```
; ****************************************************************************
; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.4
; ----------------------------------------------------------------------------
; Last Update: 18/04/2021
; ----------------------------------------------------------------------------
; Beginning: 04/01/2016
; ----------------------------------------------------------------------------
; Assembler: NASM version 2.15 (trdos386.s)
; ----------------------------------------------------------------------------
; Turkish Rational DOS
; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
;
; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
; unix386.s (03/01/2016)
;
; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
; TRDOS2.ASM (09/11/2011)
;
; Derived from 'IBM PC-XT-286' BIOS source code (1986)
; ****************************************************************************
; nasm trdos386.s -l trdos386.txt -o TRDOS386.SYS

KLOAD equ 10000h ; Kernel loading address
      ; NOTE: Retro UNIX 8086 v1 boot code loads kernel at 1000h:0000h
KCODE equ 08h      ; Code segment descriptor (ring 0)
KDATA equ 10h      ; Data segment descriptor (ring 0)
; 19/03/2015
UCODE equ 1Bh ; 18h + 3h  (ring 3)
UDATA equ 23h ; 20h + 3h  (ring 3)
; 24/03/2015
TSS   equ 28h      ; Task state segment descriptor (ring 0)
; 19/03/2015
CORE  equ 400000h  ; Start of USER's virtual/linear address space
                   ; (at the end of the 1st 4MB)
ECORE equ 0FFC00000h ; End of USER's virtual address space (4GB - 4MB)
                   ; ULIMIT = (ECORE/4096) - 1 = 0FFBFFh (in GDT)
;; 27/12/2013
;KEND equ KLOAD + 65536 ; (28/12/2013) (end of kernel space)
; 04/07/2016
KEND    equ KERNELFSIZE + KLOAD

; IBM PC/AT BIOS ----- 10/06/85 (postequ.inc)
;--------- CMOS TABLE LOCATION ADDRESS'S ---------------------------------
CMOS_SECONDS       EQU    00H          ; SECONDS (BCD)
CMOS_SEC_ALARM     EQU    01H          ; SECONDS ALARM (BCD)
CMOS_MINUTES       EQU    02H          ; MINUTES (BCD)
CMOS_MIN_ALARM     EQU    03H          ; MINUTES ALARM (BCD)
CMOS_HOURS         EQU    04H          ; HOURS (BCD
CMOS_HR_ALARM      EQU    005H         ; HOURS ALARM   (BCD)
CMOS_DAY_WEEK      EQU    06H          ; DAY OF THE WEEK  (BCD)
CMOS_DAY_MONTH     EQU    07H          ; DAY OF THE MONTH (BCD)
CMOS_MONTH         EQU    08H          ; MONTH (BCD)
CMOS_YEAR          EQU    09H          ; YEAR (TWO DIGITS) (BCD)
CMOS_CENTURY       EQU    32H          ; DATE CENTURY BYTE (BCD)
CMOS_REG_A         EQU    0AH          ; STATUS REGISTER A
CMOS_REG_B         EQU    00BH         ; STATUS REGISTER B  ALARM
CMOS_REG_C         EQU    00CH         ; STATUS REGISTER C  FLAGS
CMOS_REG_D         EQU    0DH          ; STATUS REGISTER D  BATTERY
CMOS_SHUT_DOWN     EQU    0FH          ; SHUTDOWN STATUS COMMAND BYTE
;--------------------------------------
;     CMOS EQUATES FOR THIS SYSTEM     ;
;----------------------------------------------------------------------------
CMOS_PORT    EQU    070H         ; I/O ADDRESS OF CMOS ADDRESS PORT
CMOS_DATA    EQU    071H         ; I/O ADDRESS OF CMOS DATA PORT
NMI          EQU    10000000B    ; DISABLE NMI INTERRUPTS MASK -
                                 ; HIGH BIT OF CMOS LOCATION ADDRESS

; Memory Allocation Table Address
; 05/11/2014
; 31/10/2014
MEM_ALLOC_TBL      equ    100000h ; Memory Allocation Table at the end of
                                  ; the 1st 1 MB memory space.
                                  ; (This address must be aligned
                                  ;  on 128 KB boundary, if it will be
                                  ;  changed later.)
                                  ; ((lower 17 bits of 32 bit M.A.T.
                                  ;   address must be ZERO)).
                                  ; ((((Reason: 32 bit allocation
                                  ;     instructions, dword steps)))
                                  ; (((byte >> 12 --> page >> 5)))
;04/11/2014
PDE_A_PRESENT      equ    1      ; Present flag for PDE
PDE_A_WRITE        equ    2      ; Writable (write permission) flag
PDE_A_USER         equ    4      ; User (non-system/kernel) page flag
;
PTE_A_PRESENT      equ    1      ; Present flag for PTE (bit 0)
PTE_A_WRITE        equ    2      ; Writable (write permission) flag (bit 1)
PTE_A_USER         equ    4      ; User (non-system/kernel) page flag (bit 2)
PTE_A_ACCESS       equ    32     ; Accessed flag (bit 5) ; 09/03/2015

; 17/02/2015 (unix386.s)
; 10/12/2014 - 30/12/2014 (0B000h -> 9000h) (dsectrm2.s)
DPT_SEGM equ 09000h  ; FDPT segment (EDD v1.1, EDD v3)
;
HD0_DPT      equ 0           ; Disk parameter table address for hd0
HD1_DPT      equ 32          ; Disk parameter table address for hd1
HD2_DPT      equ 64          ; Disk parameter table address for hd2
HD3_DPT      equ 96          ; Disk parameter table address for hd3

; 15/11/2020
VBE3INFOSEG equ 97E0h ; 512 bytes before Video_Pg_Backup
; 15/12/2020
VBE3MODEINFOSEG equ 97C0h ; 512 bytes before VBE3INFOBLOCK

; 29/11/2020
```

```
                                VBE3INFOBLOCK equ 97E00h ; linear address (512 bytes)
                                VBE3MODEINFOBLOCK equ 97C00h ; linear address (256 bytes)
                                VBE3SAVERESTOREBLOCK equ 97600h ; linear address (2048 bytes)
                                VBE3CRTCINFOBLOCK equ 97D80h ; linear address (64 bytes) ; 17/01/2021
                                VBE3BIOSDATABLOCK equ 97000h ; linear address (1536 bytes)
                                VBE3STACKADDR equ 96000h ; linear address (1024 bytes)
                                ; VBE3 32 bit Protected Mode Interface (16 bit) Selectors (in GDT)
                                VBE3CS equ 30h ; _vbe3_CS:
                                VBE3BDS equ 38h ; _vbe3_BDS:
                                VBE3A000 equ 40h ; _A0000Sel:
                                VBE3B000 equ 48h ; _B0000Sel:
                                VBE3B800 equ 50h ; _B8000Sel:
                                VBE3DS equ 58h ; _vbe3_DS:
                                VBE3SS equ 60h ; _vbe3_SS:
                                VBE3ES equ 68h ; _vbe3_ES:
                                KCODE16 equ 70h ; _16bit_CS:
                                ; 14/01/2021
                                ; 06/12/2020
                                VBE3VIDEOSTATE equ 95800h ; 2048 bytes
                                ; 05/01/2021
                                VGAFONT16USER equ 94000h ; 8x16 pixels user font (256 chars)
                                                  ; (reserved/allocated font space: 4096 bytes)

                                VGAFONT8USER equ 95000h   ; 8x8 pixels user font (256 chars)
                                                 ; (reserved/allocated font space: 2048 bytes)
                                ; 17/01/2021
                                ; temporary (initial) location for EDID information
                                VBE3EDIDINFOBLOCK equ 97D00h ; linear address (128 bytes)

                                ; FDPT (Phoenix, Enhanced Disk Drive Specification v1.1, v3.0)
                                ;      (HDPT: Programmer's Guide to the AMIBIOS, 1993)
                                ;
                                FDPT_CYLS    equ 0 ; 1 word, number of cylinders
                                FDPT_HDS     equ 2 ; 1 byte, number of heads
                                FDPT_TT      equ 3 ; 1 byte, A0h = translated FDPT with logical values
                                                   ; otherwise it is standard FDPT with physical values
                                FDPT_PCMP    equ 5 ; 1 word, starting write precompensation cylinder
                                                   ; (obsolete for IDE/ATA drives)
                                FDPT_CB      equ 8 ; 1 byte, drive control byte
                                                    ; Bits 7-6 : Enable or disable retries (00h = enable)
                                                    ; Bit 5      : 1 = Defect map is located at last cyl. + 1
                                                    ; Bit 4 : Reserved. Always 0
                                                    ; Bit 3 : Set to 1 if more than 8 heads
                                                    ; Bit 2-0 : Reserved. Always 0
                                FDPT_LZ      equ 12 ; 1 word, landing zone (obsolete for IDE/ATA drives)
                                FDPT_SPT     equ 14 ; 1 byte, sectors per track

                                ; Floppy Drive Parameters Table (Programmer's Guide to the AMIBIOS, 1993)
                                ; (11 bytes long) will be used by diskette handler/bios
                                ; which is derived from IBM PC-AT BIOS (DISKETTE.ASM, 21/04/1986).

                                ; 01/02/2016
                                Logical_DOSDisks equ 90000h + 100h ; 26*256 = 6656 bytes
                                Directory_Buffer equ 80000h ; max = 64K Bytes
                                FAT_Buffer   equ 91C00h ; 1536 bytes (3 sectors)
                                ; 15/02/2016
                                Cluster_Buffer equ 70000h ; max = 64K Bytes ; buffer for file read & write
                                ; 11/04/2016
                                Env_Page:    equ 93000h ; 512 bytes (4096 bytes)
                                Env_Page_Size equ 512    ; (4096 bytes)
                                ; 30/07/2016
                                Video_Pg_Backup equ 98000h ; Mode 3h, video page backup (32K, 8 pages)

                                ; 29/11/2020
                                ; Free/Reserved memory blocks (in 1st 1MB): 93200h to 96000h (available)
                                ; 06/12/2020
                                ; Free/Reserved memory blocks (in 1st 1MB): 93200h to 95800h (available)

                                ; 15/12/2020
                                LFB_ADDR     equ LFB_Info+LFBINFO.LFB_addr
                                LFB_SIZE     equ LFB_Info+LFBINFO.LFB_size

                                [BITS 16]       ; We need 16-bit intructions for Real mode

                                [ORG 0]
                                      ; 12/11/2014
                                      ; Save boot drive number (that is default root drive)
00000000 8816[EA68]                   mov   [boot_drv], dl ; physical drv number

                                      ; Determine installed memory
                                      ; 31/10/2014
                                      ;
00000004 B801E8                       mov   ax, 0E801h ; Get memory size
00000007 CD15                         int   15h       ; for large configurations
00000009 7308                         jnc   short chk_ms
0000000B B488                         mov   ah, 88h    ; Get extended memory size
0000000D CD15                         int   15h
                                      ;
                                      ;mov  al, 17h      ; Extended memory (1K blocks) low byte
                                      ;out  70h, al ; select CMOS register
                                      ;in   al, 71h ; read data (1 byte)
                                      ;mov  cl, al
                                      ;mov  al, 18h ; Extended memory (1K blocks) high byte
                                      ;out  70h, al ; select CMOS register
                                      ;in   al, 71h ; read data (1 byte)
                                      ;mov  ch, al
                                      ;
0000000F 89C1                         mov   cx, ax
00000011 31D2                         xor   dx, dx
                                chk_ms:
00000013 890E[E668]                   mov   [mem_1m_1k], cx
00000017 8916[E868]                   mov   [mem_16m_64k], dx
                                      ; 05/11/2014
                                      ;and  dx, dx
                                      ;jz   short L2
```

```
0000001B 81F90004                       cmp     cx, 1024
                                        ;jnb   short L0
0000001F 7351                           jnb    short V0 ; 14/11/2020
                                               ; insufficient memory_error
                                               ; Minimum 2 MB memory is needed...
                                        ; 05/11/2014
                                        ; (real mode error printing)
00000021 FB                             sti
00000022 BE[3600]                       mov    si, msg_out_of_memory
00000025 BB0700                         mov    bx, 7
00000028 B40E                           mov    ah, 0Eh      ; write tty
                                oom_1:
0000002A AC                             lodsb
0000002B 08C0                           or     al, al
0000002D 7404                           jz     short oom_2
0000002F CD10                           int    10h
00000031 EBF7                           jmp    short oom_1
                                oom_2:
00000033 F4                             hlt
00000034 EBFD                           jmp    short oom_2

                                ; 20/02/2017
                                ; 05/11/2014
                                msg_out_of_memory:
00000036 070D0A                         db     07h, 0Dh, 0Ah
00000039 496E73756666696369-            db      'Insufficient memory !'
00000042 656E74206D656D6F72-
0000004B 792021
0000004E 0D0A                           db     0Dh, 0Ah
                                _int13h_48h_buffer: ; 07/07/2016
00000050 284D696E696D756D20-            db     '(Minimum 2MB memory is needed.)'
00000059 324D42206D656D6F72-
00000062 79206973206E656564-
0000006B 65642E29
0000006F 0D0A00                         db     0Dh, 0Ah, 0
                                V0:
                                        ; 15/12/2020
00000072 8B36[E868]                     mov    si, [mem_16m_64k]
00000076 8936[D70E]                     mov    [real_mem_16m_64k], si
                                        ; 15/11/2020
                                        ; 14/11/2020 (TRDOS 386 v2.0.3)
                                        ; check VESA (VBE) VIDEO BIOS version

0000007A B8034F                         mov    ax, 4F03h  ; Return current VBE mode
0000007D CD10                           int    10h
0000007F 83F84F                         cmp    ax, 004Fh  ; successful (vbe) function call
00000082 7567                           jne    short L0   ; not a VESA VBE compatible bios

                                        ;mov   ah, 3
                                        ;;jmp  short v1

                                        ; 15/11/2020
00000084 BBE097                         mov    bx, VBE3INFOSEG  ; 97E0h for current version
00000087 8EC3                           mov    es, bx
00000089 31FF                           xor    di, di
0000008B 2666C70556424532               mov    dword [es:di], 'VBE2' ; request VESA VBE3 info
                                               ; es:di = buffer address (512 bytes)
                                        ;mov   ax, 4F00h ; Return VBE controller information
00000093 86C4                           xchg   al, ah
00000095 CD10                           int    10h

                                        ; dx = cs
                                        ; es = VBE3INFOSEG (97E0h)
                                        ; di = 0
                                        ; ss = (endofkernelfile/16)+16
                                        ; sp = 0FFFEh

00000097 83F84F                         cmp    ax, 004Fh
0000009A 754D                           jne    short V1 ; old vga bios (not VESA compatible)

                                        ; 15/11/2020
0000009C 2666813D56455341               cmp    dword [es:di], 'VESA'
000000A4 7543                           jne    short V1

                                        ;mov   ax, [es:di+4]
                                        ;      ; ax = vbe version in BCD format (0200h or 0300h)
                                        ;mov   [vbe3], ah ; version number (major)

                                        ; 15/11/2020
000000A6 268A4505                       mov    al, [es:di+5]
                                               ; al = high byte of VBE version number (02h or 03h)

000000AA A2[4E09]                       mov    [vbe3], al ; version number (major)
                                                          ; 02h or 03h is expected
                                        ; 17/01/2021
                                        ; Read EDID
000000AD B301                           mov    bl, 01h      ; Read EDID
000000AF 31C9                           xor    cx, cx ; Controller unit number
                                                      ; (00 = primary controller)
000000B1 31D2                           xor    dx, dx ; EDID block number = 0
000000B3 B8C097                         mov    ax, VBE3MODEINFOSEG  ; 97C0h for current version
000000B6 8EC0                           mov    es, ax
000000B8 BF0001                         mov    di, VBE3EDIDINFOBLOCK - VBE3MODEINFOBLOCK
                                        ; es:di = temporary address of 128 bytes EDID information
000000BB B8154F                         mov    ax, 4F15h ; VBE/DDC Services
000000BE CD10                           int    10h
                                        ;cmp   ax, 4Fh
                                        ;jne   short v2
000000C0 A2[6D42]                       mov    [edid], al ; 4Fh > 0
                                ;V2:
                                        ; 17/01/2021
000000C3 31FF                           xor    di, di
                                        ; 15/12/2020
                                        ; Get linear frame buffer info (for VESA VBE mode 118h)
                                        ;mov   si, VBE3MODEINFOSEG  ; 97C0h for current version
```

```
                                        ;mov   es, si
                                        ; di = 0
000000C5 B91841                         mov    cx, 04118h  ; 1024*768, 24 bpp, LFB
000000C8 B8014F                         mov    ax, 4F01h ; Return VBE mode information
000000CB CD10                           int    10h
                                        ;cmp   ax, 4Fh
                                        ;jne   short V1
                                        ; 19/12/2020
                                        ;mov   si, [es:di+MODEINFO.PhysBasePtr+2]
                                                   ; hw of LFB base address
                                        ; MODEINFO structure starts from offset -2
000000CD 268B752A                       mov    si, [es:di+MODEINFO.PhysBasePtr] ; hw of LFB addr
000000D1 8936[D90E]                     mov    [def_LFB_addr], si ; k_LFB_size = 3145728 bytes
000000D5 81EE0001                       sub    si, 256

                                        ; 15/12/2020
                                        ; check memory and decrease it to 3.5 GB if it is 4GB
                                        ; (reserve upper memory for LFB)
000000D9 8B3E[E868]                     mov    di, [mem_16m_64k]
000000DD 893E[D70E]                     mov    [real_mem_16m_64k], di

000000E1 39F7                           cmp    di, si
000000E3 7604                           jna    short V1

000000E5 8936[E868]                     mov    [mem_16m_64k], si

                                        ; VESA VBE3 video hardware
                                        ; (example: NVIDIA GEFORCE FX550, 256 MB)
                                        ; uses upper memory from 0D0000000h to 0DFFFFFFFh

                                        ;;cmp  di, 0CF00h ; 3328 MB - 16MB
                                        ;jna   short V1  ; <= 3328 MB memory, it is not required
                                                         ; decrease
                                              ;cmp   al, 3
                                              ;jb    short V2
                                              ; VESA VBE 3
                                              ;mov   word [mem_16m_64k], 0CF00h ; 3328 MB - 16MB
                                              ;jmp   short V1
                                        ;V2:
                                              ; VESA VBE 2
                                              ; Check Bochs/Qemu/VirtualBox Emulator
                                              ; LFB base address: 0E0000000h
                                              ;sub   ax, ax ; 0
                                              ;mov   dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
                                              ;out   dx, ax ; VBE_DISPI_INDEX_ID register
                                              ;inc   dx
                                              ;in    ax, dx
                                              ;and   al, 0F0h
                                              ;cmp   ax, 0B0C0h
                                              ;jne   short V1
                                              ;
                                              ; BOCHS/QEMU/VIRTUALBOX
                                              ;mov   word [mem_16m_64k], 0DF00h ; 3584 MB - 16MB
                                        V1:
000000E9 1E                                   push   ds
000000EA 07                                   pop    es ; restore extra data segment
                                        L0:

                                        %include 'diskinit.s' ; 07/03/2015
                                    <1> ; ****************************************************************************
                                    <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.4 - diskinit.s
                                    <1> ; ----------------------------------------------------------------------------
                                    <1> ; Last Update: 18/04/2021
                                    <1> ; ----------------------------------------------------------------------------
                                    <1> ; Beginning: 24/01/2016
                                    <1> ; ----------------------------------------------------------------------------
                                    <1> ; Assembler: NASM version 2.11 (trdos386.s)
                                    <1> ; ----------------------------------------------------------------------------
                                    <1> ; Turkish Rational DOS
                                    <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
                                    <1> ;
                                    <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
                                    <1> ; diskinit.inc (10/07/2015)
                                    <1> ;
                                    <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
                                    <1> ; ****************************************************************************
                                    <1>
                                    <1> ; Retro UNIX 386 v1 Kernel - DISKINIT.INC
                                    <1> ; Last Modification: 10/07/2015
                                    <1>
                                    <1> ; DISK I/O SYSTEM INITIALIZATION - Erdogan Tan (Retro UNIX 386 v1 project)
                                    <1>
                                    <1> ; ///////// DISK I/O SYSTEM STRUCTURE INITIALIZATION ///////////////
                                    <1>
                                    <1>       ; 10/12/2014 - 02/02/2015 - dsectrm2.s
                                    <1> ;L0:
                                    <1>       ; 12/11/2014 (Retro UNIX 386 v1 - beginning)
                                    <1>       ; Detecting disk drives... (by help of ROM-BIOS)
000000EB BA7F00                     <1>       mov    dx, 7Fh
                                    <1> L1:
000000EE FEC2                       <1>       inc    dl
000000F0 B441                       <1>       mov    ah, 41h ; Check extensions present
                                    <1>                       ; Phoenix EDD v1.1 - EDD v3
000000F2 BBAA55                     <1>       mov    bx, 55AAh
000000F5 CD13                       <1>       int    13h
000000F7 721A                       <1>       jc     short L2
                                    <1>
000000F9 81FB55AA                   <1>       cmp    bx, 0AA55h
000000FD 7514                       <1>       jne    short L2
000000FF FE06[ED68]                 <1>       inc    byte [hdc]   ; count of hard disks (EDD present)
00000103 8816[EC68]                 <1>       mov    [last_drv], dl  ; last hard disk number
00000107 BB[7068]                   <1>       mov    bx, hd0_type - 80h
0000010A 01D3                       <1>       add    bx, dx
0000010C 880F                       <1>       mov    [bx], cl ; Interface support bit map in CX
                                    <1>                       ; Bit 0 - 1, Fixed disk access subset ready
```

```
                                <1>                     ; Bit 1 - 1, Drv locking and ejecting ready
                                <1>                     ; Bit 2 - 1, Enhanced Disk Drive Support
                                <1>                         ;            (EDD) ready (DPTE ready)
                                <1>                     ; Bit 3 - 1, 64bit extensions are present
                                <1>                         ;            (EDD-3)
                                <1>                     ; Bit 4 to 15 - 0, Reserved
0000010E 80FA83                 <1>         cmp    dl, 83h       ; drive number < 83h
00000111 72DB                   <1>         jb     short L1
                                <1> L2:
                                <1>         ; 23/11/2014
                                <1>         ; 19/11/2014
00000113 30D2                   <1>         xor    dl, dl  ; 0
                                <1>         ; 04/02/2016 (esi -> si)
00000115 BE[EE68]               <1>         mov    si, fd0_type
                                <1> L3:
                                <1>         ; 14/01/2015
00000118 8816[EB68]             <1>         mov    [drv], dl
                                <1>         ;
0000011C B408                   <1>         mov    ah, 08h ; Return drive parameters
0000011E CD13                   <1>         int    13h
00000120 7210                   <1>         jc     short L4
                                <1>                ; BL = drive type (for floppy drives)
                                <1>                ; DL = number of floppy drives
                                <1>                ;
                                <1>                ; ES:DI = Address of DPT from BIOS
                                <1>                ;
00000122 881C                   <1>         mov    [si], bl ;  Drive type
                                <1>                       ; 4 = 1.44 MB, 80 track, 3 1/2"
                                <1>         ; 14/01/2015
00000124 E8BD01                 <1>         call   set_disk_parms
                                <1>         ; 10/12/2014
00000127 81FE[EE68]             <1>         cmp    si, fd0_type
0000012B 7705                   <1>         ja     short L4
0000012D 46                     <1>         inc    si ; fd1_type
0000012E B201                   <1>         mov    dl, 1
00000130 EBE6                   <1>         jmp    short L3
                                <1> L4:
                                <1>         ; Older BIOS (INT 13h, AH = 48h is not available)
00000132 B27F                   <1>         mov    dl, 7Fh
                                <1>         ; 24/12/2014 (Temporary)
00000134 803E[ED68]00           <1>         cmp    byte [hdc], 0 ; EDD present or not ?
00000139 0F879000               <1>         ja     L10       ; yes, all fixed disk operations
                                <1>                          ; will be performed according to
                                <1>                          ; present EDD specification
                                <1> L6:
0000013D FEC2                   <1>         inc    dl
0000013F 8816[EB68]             <1>         mov    [drv], dl
00000143 8816[EC68]             <1>         mov    [last_drv], dl ; 14/01/2015
00000147 B408                   <1>         mov    ah, 08h ; Return drive parameters
00000149 CD13                   <1>         int    13h    ; (conventional function)
0000014B 0F828701               <1>         jc     L13    ; fixed disk drive not ready
0000014F 8816[ED68]             <1>         mov    [hdc], dl ; number of drives
                                <1>         ;; 14/01/2013
                                <1>         ;;push cx
00000153 E88E01                 <1>         call   set_disk_parms
                                <1>         ;;pop  cx
                                <1>         ;
                                <1>         ;;and  cl, 3Fh       ; sectors per track (bits 0-6)
00000156 8A16[EB68]             <1>         mov    dl, [drv]
0000015A BB0401                 <1>         mov    bx, 65*4 ; hd0 parameters table (INT 41h)
0000015D 80FA80                 <1>         cmp    dl, 80h
00000160 7603                   <1>         jna    short L7
00000162 83C314                 <1>         add    bx, 5*4       ; hd1 parameters table (INT 46h)
                                <1> L7:
00000165 31C0                   <1>         xor    ax, ax
00000167 8ED8                   <1>         mov    ds, ax
00000169 8B37                   <1>         mov    si, [bx]
0000016B 8B4702                 <1>         mov    ax, [bx+2]
0000016E 8ED8                   <1>         mov    ds, ax
00000170 3A4C0E                 <1>         cmp    cl, [si+FDPT_SPT] ; sectors per track
00000173 0F855B01               <1>         jne    L12 ; invalid FDPT
00000177 BF0000                 <1>         mov    di, HD0_DPT
0000017A 80FA80                 <1>         cmp    dl, 80h
0000017D 7603                   <1>         jna    short L8
0000017F BF2000                 <1>         mov    di, HD1_DPT
                                <1> L8:
                                <1>         ; 30/12/2014
00000182 B80090                 <1>         mov    ax, DPT_SEGM
00000185 8EC0                   <1>         mov    es, ax
                                <1>         ; 24/12/2014
00000187 B90800                 <1>         mov    cx, 8
0000018A F3A5                   <1>         rep    movsw  ; copy 16 bytes to the kernel's DPT location
0000018C 8CC8                   <1>         mov    ax, cs
0000018E 8ED8                   <1>         mov    ds, ax
                                <1>         ; 02/02/2015
00000190 8A0E[EB68]             <1>         mov    cl, [drv]
00000194 88CB                   <1>         mov    bl, cl
00000196 B8F001                 <1>         mov    ax, 1F0h
00000199 80E301                 <1>         and    bl, 1
0000019C 7406                   <1>         jz     short L9
0000019E C0E304                 <1>         shl    bl, 4
000001A1 2D8000                 <1>         sub    ax, 1F0h-170h
                                <1> L9:
000001A4 AB                     <1>         stosw  ; I/O PORT Base Address (1F0h, 170h)
000001A5 050602                 <1>         add    ax, 206h
000001A8 AB                     <1>         stosw  ; CONTROL PORT Address (3F6h, 376h)
000001A9 88D8                   <1>         mov    al, bl
000001AB 04A0                   <1>         add    al, 0A0h
000001AD AA                     <1>         stosb  ; Device/Head Register upper nibble
                                <1>         ;
000001AE FE06[EB68]             <1>         inc    byte [drv]
000001B2 BB[7068]               <1>         mov    bx, hd0_type - 80h
000001B5 01CB                   <1>         add    bx, cx
000001B7 800F80                 <1>         or     byte [bx], 80h  ; present sign (when lower nibble is 0)
000001BA A0[ED68]               <1>         mov    al, [hdc]
```

```
000001BD FEC8               <1>       dec    al
000001BF 0F841301           <1>       jz     L13
000001C3 80FA80             <1>       cmp    dl, 80h
000001C6 0F8673FF           <1>       jna    L6
000001CA E90901             <1>       jmp    L13
                            <1> L10:
000001CD FEC2               <1>       inc    dl
                            <1>       ; 25/12/2014
000001CF 8816[EB68]         <1>       mov    [drv], dl
000001D3 B408               <1>       mov    ah, 08h ; Return drive parameters
000001D5 CD13               <1>       int    13h    ; (conventional function)
000001D7 0F82FB00           <1>       jc     L13
                            <1>       ; 14/01/2015
000001DB 8A16[EB68]         <1>       mov    dl, [drv]
000001DF 52                 <1>       push   dx
000001E0 51                 <1>       push   cx
000001E1 E80001             <1>       call   set_disk_parms
000001E4 59                 <1>       pop    cx
000001E5 5A                 <1>       pop    dx
                            <1>       ; 06/07/2016 (BugFix for >64K kernel files)
                            <1>       ; 04/02/2016 (esi -> si)
                            <1>       ;mov   si, _end ; 30 byte temporary buffer address
                            <1>       ;              ; at the '_end' of kernel.
                            <1>       ;mov   word [si], 30
                            <1>       ; 06/07/2016
000001E6 BE[5000]           <1>       mov    si, _int13h_48h_buffer
                            <1>       ; 09/07/2016
000001E9 B81E00             <1>       mov    ax, 001Eh
000001EC 8824               <1>       mov    [si], ah ; 0
000001EE 46                 <1>       inc    si
000001EF 8904               <1>       mov    word [si], ax
                            <1>       ; word [si] = 30
                            <1>       ;
000001F1 B448               <1>       mov    ah, 48h       ; Get drive parameters (EDD function)
000001F3 CD13               <1>       int    13h
000001F5 0F82DD00           <1>       jc     L13
                            <1>       ; 04/02/2016 (ebx -> bx)
                            <1>       ; 14/01/2015
000001F9 28FF               <1>       sub    bh, bh
000001FB 88D3               <1>       mov    bl, dl
000001FD 80EB80             <1>       sub    bl, 80h
00000200 81C3[F068]         <1>       add    bx, hd0_type
00000204 8A07               <1>       mov    al, [bx]
00000206 0C80               <1>       or     al, 80h
00000208 8807               <1>       mov    [bx], al
0000020A 81EB[EE68]         <1>       sub    bx, hd0_type - 2 ; 15/01/2015
0000020E 81C3[3A69]         <1>       add    bx, drv.status
00000212 8807               <1>       mov    [bx], al
                            <1>       ; 04/02/2016 (eax -> ax)
00000214 8B4410             <1>       mov    ax, [si+16]
00000217 854412             <1>       test   ax, [si+18]
0000021A 7413               <1>       jz     short L10_A0h
                            <1>                    ; 'CHS only' disks on EDD system
                            <1>                    ;  are reported with ZERO disk size
0000021C 81EB[3A69]         <1>       sub    bx, drv.status
00000220 C1E302             <1>       shl    bx, 2
00000223 81C3[1E69]         <1>       add    bx, drv.size ; disk size (in sectors)
00000227 8907               <1>       mov    [bx], ax
00000229 8B4412             <1>       mov    ax, [si+18]
                            <1>       ; 18/04/2021
0000022C 894702             <1>       mov    [bx+2], ax
                            <1>
                            <1> L10_A0h: ; Jump here to fix a ZERO (LBA) disk size problem
                            <1>        ; for CHS disks (28/02/2015)
                            <1>       ; 30/12/2014
0000022F BF0000             <1>       mov    di, HD0_DPT
00000232 88D0               <1>       mov    al, dl
00000234 83E003             <1>       and    ax, 3
00000237 C0E005             <1>       shl    al, 5 ; *32
0000023A 01C7               <1>       add    di, ax
0000023C B80090             <1>       mov    ax, DPT_SEGM
0000023F 8EC0               <1>       mov    es, ax
                            <1>       ;
00000241 88E8               <1>       mov    al, ch ; max. cylinder number (bits 0-7)
00000243 88CC               <1>       mov    ah, cl
00000245 C0EC06             <1>       shr    ah, 6  ; max. cylinder number (bits 8-9)
00000248 40                 <1>       inc    ax     ; logical cylinders (limit 1024)
00000249 AB                 <1>       stosw
0000024A 88F0               <1>       mov    al, dh ; max. head number
0000024C FEC0               <1>       inc    al
0000024E AA                 <1>       stosb         ; logical heads (limits 256)
0000024F B0A0               <1>       mov    al, 0A0h ; Indicates translated table
00000251 AA                 <1>       stosb
00000252 8A440C             <1>       mov    al, [si+12]
00000255 AA                 <1>       stosb         ; physical sectors per track
00000256 31C0               <1>       xor    ax, ax
                            <1>       ;dec   ax      ; 02/01/2015
00000258 AB                 <1>       stosw         ; precompensation (obsolete)
                            <1>       ;xor   al, al  ; 02/01/2015
00000259 AA                 <1>       stosb         ; reserved
0000025A B008               <1>       mov    al, 8   ; drive control byte
                            <1>                      ; (do not disable retries,
                            <1>                     ; more than 8 heads)
0000025C AA                 <1>       stosb
0000025D 8B4404             <1>       mov    ax, [si+4]
00000260 AB                 <1>       stosw         ; physical number of cylinders
                            <1>       ;push  ax      ; 02/01/2015
00000261 8A4408             <1>       mov    al, [si+8]
00000264 AA                 <1>       stosb         ; physical num. of heads (limit 16)
00000265 29C0               <1>       sub    ax, ax
                            <1>       ;pop   ax      ; 02/01/2015
00000267 AB                 <1>       stosw         ; landing zone (obsolete)
00000268 88C8               <1>       mov    al, cl  ; logical sectors per track (limit 63)
0000026A 243F               <1>       and    al, 3Fh
0000026C AA                 <1>       stosb
```

```
                        <1>       ;sub   al, al  ; checksum
                        <1>       ;stosb
                        <1>       ;
0000026D 83C61A         <1>       add    si, 26   ; (BIOS) DPTE address pointer
00000270 AD             <1>       lodsw
00000271 50             <1>       push   ax      ; (BIOS) DPTE offset
00000272 AD             <1>       lodsw
00000273 50             <1>       push   ax      ; (BIOS) DPTE segment
                        <1>       ;
                        <1>       ; checksum calculation
00000274 89FE           <1>       mov    si, di
00000276 06             <1>       push   es
00000277 1F             <1>       pop    ds
                        <1>       ;mov   cx, 16
00000278 B90F00         <1>       mov    cx, 15
0000027B 29CE           <1>       sub    si, cx
0000027D 30E4           <1>       xor    ah, ah
                        <1>       ;del   cl
                        <1> L11:
0000027F AC             <1>       lodsb
00000280 00C4           <1>       add    ah, al
00000282 E2FB           <1>       loop   L11
                        <1>       ;
00000284 88E0           <1>       mov    al, ah
00000286 F6D8           <1>       neg    al     ; -x+x = 0
00000288 AA             <1>       stosb         ; put checksum in byte 15 of the tbl
                        <1>       ;
00000289 1F             <1>       pop    ds     ; (BIOS) DPTE segment
0000028A 5E             <1>       pop    si     ; (BIOS) DPTE offset
                        <1>       ;
                        <1>       ; 23/02/2015
0000028B 57             <1>       push   di
                        <1>       ; ES:DI points to DPTE (FDPTE) location
                        <1>       ;mov   cx, 8
0000028C B108           <1>       mov    cl, 8
0000028E F3A5           <1>       rep    movsw
                        <1>       ;
                        <1>       ; 23/02/2015
                        <1>       ; (P)ATA drive and LBA validation
                        <1>       ; (invalidating SATA drives and setting
                        <1>       ; CHS type I/O for old type fixed disks)
00000290 5B             <1>       pop    bx
00000291 8CC8           <1>       mov    ax, cs
00000293 8ED8           <1>       mov    ds, ax
00000295 268B07         <1>       mov    ax, [es:bx]
00000298 3DF001         <1>       cmp    ax, 1F0h
0000029B 7418           <1>       je     short L11a
0000029D 3D7001         <1>       cmp    ax, 170h
000002A0 7413           <1>       je     short L11a
                        <1>       ; invalidation
                        <1>       ; (because base port address is not 1F0h or 170h)
000002A2 30FF           <1>       xor    bh, bh
000002A4 88D3           <1>       mov    bl, dl
000002A6 80EB80         <1>       sub    bl, 80h
000002A9 C687[F068]00   <1>       mov    byte [bx+hd0_type], 0 ; not a valid disk drive !
000002AE 808F[3C69]F0   <1>       or    byte [bx+drv.status+2], 0F0h ; (failure sign)
000002B3 EB14           <1>       jmp    short L11b
                        <1> L11a:
                        <1>       ; LBA validation
000002B5 268A4704       <1>       mov    al, [es:bx+4] ; Head register upper nibble
000002B9 A840           <1>       test   al, 40h ; LBA bit (bit 6)
000002BB 750C           <1>       jnz    short L11b ; LBA type I/O is OK! (E0h or F0h)
                        <1>       ; force CHS type I/O for this drive (A0h or B0h)
000002BD 28FF           <1>       sub    bh, bh
000002BF 88D3           <1>       mov    bl, dl
000002C1 80EB80         <1>       sub    bl, 80h ; 26/02/2015
000002C4 80A7[3C69]FE   <1>       and   byte [bx+drv.status+2], 0FEh ; clear bit 0
                        <1>                                ; bit 0 = LBA ready bit
                        <1>       ; 'diskio' procedure will check this bit !
                        <1> L11b:
000002C9 3A16[EC68]     <1>       cmp    dl, [last_drv] ; 25/12/2014
000002CD 7307           <1>       jnb    short L13
000002CF E9FBFE         <1>       jmp    L10
                        <1> L12:
                        <1>       ; Restore data registers
000002D2 8CC8           <1>       mov    ax, cs
000002D4 8ED8           <1>       mov    ds, ax
                        <1> L13:
                        <1>       ; 13/12/2014
000002D6 0E             <1>       push   cs
000002D7 07             <1>       pop    es
                        <1> L14:
000002D8 B411           <1>       mov    ah, 11h
000002DA CD16           <1>       int    16h
000002DC 7466           <1>       jz     short L16 ; no keys in keyboard buffer
000002DE B010           <1>       mov    al, 10h
000002E0 CD16           <1>       int    16h
000002E2 EBF4           <1>       jmp    short L14
                        <1>
                        <1> set_disk_parms:
                        <1>       ; 04/02/2016 (ebx -> bx)
                        <1>       ; 10/07/2015
                        <1>       ; 14/01/2015
                        <1>       ;push  bx
000002E4 28FF           <1>       sub    bh, bh
000002E6 8A1E[EB68]     <1>       mov    bl, [drv]
000002EA 80FB80         <1>       cmp    bl, 80h
000002ED 7203           <1>       jb     short sdp0
000002EF 80EB7E         <1>       sub    bl, 7Eh
                        <1> sdp0:
000002F2 81C3[3A69]     <1>       add    bx, drv.status
000002F6 C60780         <1>       mov    byte [bx], 80h ; 'Present' flag
                        <1>       ;
000002F9 88E8           <1>       mov    al, ch ; last cylinder (bits 0-7)
000002FB 88CC           <1>       mov    ah, cl ;
```

```
731 000002FD C0EC06             <1>     shr    ah, 6  ; last cylinder (bits 8-9)
732 00000300 81EB[3A69]         <1>     sub    bx, drv.status
733 00000304 D0E3               <1>     shl    bl, 1
734 00000306 81C3[F468]         <1>     add    bx, drv.cylinders
735 0000030A 40                 <1>     inc    ax  ; convert max. cyl number to cyl count
736 0000030B 8907               <1>     mov    [bx], ax
737 0000030D 50                 <1>     push   ax ; ** cylinders
738 0000030E 81EB[F468]         <1>     sub    bx, drv.cylinders
739 00000312 81C3[0269]         <1>     add    bx, drv.heads
740 00000316 30E4               <1>     xor    ah, ah
741 00000318 88F0               <1>     mov    al, dh ; heads
742 0000031A 40                 <1>     inc    ax
743 0000031B 8907               <1>     mov    [bx], ax
744 0000031D 81EB[0269]         <1>     sub    bx, drv.heads
745 00000321 81C3[1069]         <1>     add    bx, drv.spt
746 00000325 30ED               <1>     xor    ch, ch
747 00000327 80E13F             <1>     and    cl, 3Fh      ; sectors (bits 0-6)
748 0000032A 890F               <1>     mov    [bx], cx
749 0000032C 81EB[1069]         <1>     sub    bx, drv.spt
750 00000330 D1E3               <1>     shl    bx, 1
751 00000332 81C3[1E69]         <1>     add    bx, drv.size ; disk size (in sectors)
752                             <1>     ; LBA size = cylinders * heads * secpertrack
753 00000336 F7E1               <1>     mul    cx
754 00000338 89C2               <1>     mov    dx, ax ; heads*spt
755 0000033A 58                 <1>     pop    ax ; ** cylinders
756 0000033B 48                 <1>     dec    ax ; 1 cylinder reserved (!?)
757 0000033C F7E2               <1>     mul    dx ; cylinders * (heads*spt)
758 0000033E 8907               <1>     mov    [bx], ax
759 00000340 895702             <1>     mov    [bx+2], dx
760                             <1>     ;
761                             <1>     ;pop   bx
762 00000343 C3                 <1>     retn
763                             <1>
764                             <1> L16:  ; 28/05/2016
370
371                                     ; 10/11/2014
372 00000344 FA                         cli    ; Disable interrupts (clear interrupt flag)
373                                     ; Reset Interrupt MASK Registers (Master&Slave)
374                                     ;mov   al, 0FFh     ; mask off all interrupts
375                                     ;out   21h, al             ; on master PIC (8259)
376                                     ;jmp   $+2  ; (delay)
377                                     ;out   0A1h, al     ; on slave PIC (8259)
378                                     ;
379                                     ; Disable NMI
380 00000345 B080                       mov    al, 80h
381 00000347 E670                       out    70h, al             ; set bit 7 to 1 for disabling NMI
382                                     ;23/02/2015
383                                     ;nop                ;
384                                     ;in    al, 71h              ; read in 71h just after writing out to 70h
385                                                         ; for preventing unknown state (!?)
386                                     ;
387                                     ; 20/08/2014
388                                     ; Moving the kernel 64 KB back (to physical address 0)
389                                     ; DS = CS = 1000h
390                                     ; 05/11/2014
391 00000349 31C0                       xor    ax, ax
392 0000034B 8EC0                       mov    es, ax ; ES = 0
393                                     ;
394                                     ; 04/07/2016 - TRDOS 386 (64K - 128K kernel)
395 0000034D 31F6                             xor    si, si
396 0000034F 31FF                       xor    di, di
397 00000351 B90040                     mov    cx, 16384
398 00000354 F366A5                     rep    movsd
399                                     ;
400 00000357 06                         push   es ; 0
401 00000358 68[5C03]                   push   L17
402 0000035B CB                         retf
403                                 L17:
404 0000035C B90010                           mov    cx, 1000h
405 0000035F 8EC1                             mov    es, cx  ; 1000h
406 00000361 01C9                             add    cx, cx
407 00000363 8ED9                             mov    ds, cx  ; 2000h
408 00000365 29F6                             sub    si, si
409 00000367 29FF                             sub    di, di
410 00000369 B90040                           mov    cx, 16384
411 0000036C F366A5                           rep    movsd
412
413                                           ; Turn off the floppy drive motor
414 0000036F BAF203                           mov   dx, 3F2h
415 00000372 EE                               out   dx, al ; 0 ; 31/12/2013
416
417                                           ; Enable access to memory above one megabyte
418                                 L18:
419 00000373 E464                             in     al, 64h
420 00000375 A802                             test   al, 2
421 00000377 75FA                             jnz    short L18
422 00000379 B0D1                             mov    al, 0D1h     ; Write output port
423 0000037B E664                             out    64h, al
424                                 L19:
425 0000037D E464                             in     al, 64h
426 0000037F A802                             test   al, 2
427 00000381 75FA                             jnz    short L19
428 00000383 B0DF                             mov    al, 0DFh     ; Enable A20 line
429 00000385 E660                             out    60h, al
430                                 ;L20:
431                                           ;
432                                           ; Load global descriptor table register
433
434                                           ;mov   ax, cs
435                                           ;mov   ds, ax
436
437 00000387 2E0F0116[5868]                   lgdt  [cs:gdtd]
438
439 0000038D 0F20C0                           mov   eax, cr0
440                                           ;or    eax, 1
```

```
00000390 40                             inc     ax
00000391 0F22C0                         mov     cr0, eax

                                        ; Jump to 32 bit code

00000394 66                             db 66h              ; Prefix for 32-bit
00000395 EA                             db 0EAh             ; Opcode for far jump
00000396 [9C030000]                     dd StartPM          ; Offset to start, 32-bit
                                                            ; (1000h:StartPM = StartPM + 10000h)
0000039A 0800                           dw KCODE            ; This is the selector for CODE32_DESCRIPTOR,
                                                            ; assuming that StartPM resides in code32

                                ; 20/02/2017


                                [BITS 32]

                                StartPM:
                                      ; Kernel Base Address = 0 ; 30/12/2013
0000039C 66B81000                     mov   ax, KDATA     ; Save data segment identifier
000003A0 8ED8                         mov   ds, ax        ; Move a valid data segment into DS register
000003A2 8EC0                         mov   es, ax        ; Move data segment into ES register
000003A4 8EE0                         mov   fs, ax        ; Move data segment into FS register
000003A6 8EE8                         mov   gs, ax        ; Move data segment into GS register
000003A8 8ED0                         mov   ss, ax        ; Move data segment into SS register
000003AA BC00000900                   mov   esp, 90000h   ; Move the stack pointer to 090000h

                                clear_bss: ; Clear uninitialized data area
                                      ; 11/03/2015
000003AF 31C0                         xor   eax, eax ; 0
000003B1 B935650000                   mov   ecx, (bss_end - bss_start)/4
                                      ;shr   ecx, 2 ; bss section is already aligned for double words
000003B6 BF[BE7D0100]                 mov   edi, bss_start
000003BB F3AB                         rep   stosd

                                memory_init:
                                      ; Initialize memory allocation table and page tables
                                      ; 18/04/2021 (TRDOS 386 v2.0.4)
                                      ; 16/11/2014
                                      ; 15/11/2014
                                      ; 07/11/2014
                                      ; 06/11/2014
                                      ; 05/11/2014
                                      ; 04/11/2014
                                      ; 31/10/2014 (Retro UNIX 386 v1 - Beginning)
                                      ;
                                ;     xor   eax, eax
                                ;     xor   ecx, ecx
000003BD B108                         mov   cl, 8
000003BF BF00001000                   mov   edi, MEM_ALLOC_TBL
000003C4 F3AB                         rep   stosd         ; clear Memory Allocation Table
                                                          ; for the first 1 MB memory
                                      ;
000003C6 668B0D[E6680000]             mov   cx, [mem_1m_1k]       ; Number of contiguous KB between
                                                          ; 1 and 16 MB, max. 3C00h = 15 MB.
000003CD 66C1E902                     shr   cx, 2         ; convert 1 KB count to 4 KB count
000003D1 890D[B0800100]               mov   [free_pages], ecx
000003D7 668B15[E8680000]             mov   dx, [mem_16m_64k]  ; Number of contiguous 64 KB blocks
                                                          ; between 16 MB and 4 GB.
000003DE 6609D2                       or    dx, dx
000003E1 7413                         jz    short mi_0
                                      ;
000003E3 6689D0                       mov   ax, dx
000003E6 C1E004                       shl   eax, 4        ; 64 KB -> 4 KB (page count)
000003E9 0105[B0800100]               add   [free_pages], eax
000003EF 0500100000                   add   eax, 4096     ; 16 MB = 4096 pages
000003F4 EB07                         jmp   short mi_1
                                mi_0:
000003F6 6689C8                       mov   ax, cx
000003F9 66050001                     add   ax, 256             ; add 256 pages for the first 1 MB
                                mi_1:
000003FD A3[AC800100]                 mov   [memory_size], eax ; Total available memory in pages
                                                          ; 1 alloc. tbl. bit = 1 memory page
                                                          ; 32 allocation bits = 32 mem. pages
                                ;
00000402 05FF7F0000             add   eax, 32767    ; 32768 memory pages per 1 M.A.T. page
00000407 C1E80F                 shr   eax, 15             ; ((32768 * x) + y) pages (y < 32768)
                                                          ;  --> x + 1 M.A.T. pages, if y > 0
                                                          ;  --> x M.A.T. pages, if y = 0
0000040A 66A3[C0800100]         mov   [mat_size], ax      ; Memory Alloc. Table Size in pages
00000410 C1E00C                 shl   eax, 12             ; 1 M.A.T. page = 4096 bytes
                                ;                         ; Max. 32 M.A.T. pages (4 GB memory)
00000413 89C3                   mov   ebx, eax      ; M.A.T. size in bytes
                                ; Set/Calculate Kernel's Page Directory Address
00000415 81C300001000           add   ebx, MEM_ALLOC_TBL
0000041B 891D[A8800100]         mov   [k_page_dir], ebx  ; Kernel's Page Directory address
                                                          ; just after the last M.A.T. page
                                ;
00000421 83E804                 sub   eax, 4        ; convert M.A.T. size to offset value
00000424 A3[B8800100]           mov   [last_page], eax   ; last page ofset in the M.A.T.
                                ;                         ; (allocation status search must be
                                                          ; stopped after here)
00000429 31C0                   xor   eax, eax
0000042B 48                     dec   eax           ; FFFFFFFFh (set all bits to 1)
                                ;push  cx
                                ; 18/04/2021
0000042C 51                     push  ecx
0000042D C1E905                 shr   ecx, 5        ; convert 1 - 16 MB page count to
                                                    ; count of 32 allocation bits
00000430 F3AB                   rep   stosd
                                ;pop   cx
                                ; 18/04/2021
00000432 59                     pop   ecx
00000433 40                     inc   eax           ; 0
00000434 80E11F                 and   cl, 31        ; remain bits
```

```
546 00000437 7412                       jz      short mi_4
547 00000439 8907                       mov     [edi], eax      ; reset
548                                 mi_2:
549 0000043B 0FAB07                     bts     [edi], eax      ; 06/11/2014
550 0000043E FEC9                       dec     cl
551 00000440 7404                       jz      short mi_3
552 00000442 FEC0                       inc     al
553 00000444 EBF5                       jmp     short mi_2
554                                 mi_3:
555 00000446 28C0                       sub     al, al          ; 0
556 00000448 83C704                     add     edi, 4          ; 15/11/2014
557                                 mi_4:
558 0000044B 6609D2                     or      dx, dx          ; check 16M to 4G memory space
559 0000044E 7421                       jz      short mi_6      ; max. 16 MB memory, no more...
560                                     ;
561 00000450 B900021000                 mov     ecx, MEM_ALLOC_TBL + 512 ; End of first 16 MB memory
562                                     ;
563 00000455 29F9                       sub     ecx, edi        ; displacement (to end of 16 MB)
564 00000457 7406                       jz      short mi_5      ; jump if EDI points to
565                                                             ;         end of first 16 MB
566 00000459 D1E9                       shr     ecx, 1          ; convert to dword count
567 0000045B D1E9                       shr     ecx, 1          ; (shift 2 bits right)
568 0000045D F3AB                       rep     stosd           ; reset all bits for reserved pages
569                                                             ; (memory hole under 16 MB)
570                                 mi_5:
571 0000045F 6689D1                     mov     cx, dx          ; count of 64 KB memory blocks
572 00000462 D1E9                       shr     ecx, 1          ; 1 alloc. dword per 128 KB memory
573 00000464 9C                         pushf                   ; 16/11/2014
574 00000465 48                         dec     eax             ; FFFFFFFFh (set all bits to 1)
575 00000466 F3AB                       rep     stosd
576 00000468 40                         inc     eax             ; 0
577 00000469 9D                         popf                    ; 16/11/2014
578 0000046A 7305                       jnc     short mi_6
579 0000046C 6648                       dec     ax              ; eax = 0000FFFFh
580 0000046E AB                         stosd
581 0000046F 6640                       inc     ax              ; 0
582                                 mi_6:
583 00000471 39DF                       cmp     edi, ebx        ; check if EDI points to
584 00000473 730A                       jnb     short mi_7      ; end of memory allocation table
585                                     ;                       ; (>= MEM_ALLOC_TBL + 4906)
586 00000475 89D9                       mov     ecx, ebx        ; end of memory allocation table
587 00000477 29F9                       sub     ecx, edi        ; convert displacement/offset
588 00000479 D1E9                       shr     ecx, 1          ; to dword count
589 0000047B D1E9                       shr     ecx, 1          ; (shift 2 bits right)
590 0000047D F3AB                       rep     stosd           ; reset all remain M.A.T. bits
591                                 mi_7:
592                                     ; Reset M.A.T. bits in M.A.T. (allocate M.A.T. pages)
593 0000047F BA00001000                 mov     edx, MEM_ALLOC_TBL
594                                     ;sub    ebx, edx        ; Mem. Alloc. Tbl. size in bytes
595                                     ;shr    ebx, 12                 ; Mem. Alloc. Tbl. size in pages
596 00000484 668B0D[C0800100]           mov     cx, [mat_size]          ; Mem. Alloc. Tbl. size in pages
597 0000048B 89D7                       mov     edi, edx
598 0000048D C1EF0F                     shr     edi, 15                 ; convert M.A.T. address to
599                                                             ; byte offset in M.A.T.
600                                                             ; (1 M.A.T. byte points to
601                                                             ;         32768 bytes)
602                                                             ; Note: MEM_ALLOC_TBL address
603                                                             ; must be aligned on 128 KB
604                                                             ; boundary!
605 00000490 01D7                       add     edi, edx        ; points to M.A.T.'s itself
606                                     ; eax = 0
607 00000492 290D[B0800100]             sub     [free_pages], ecx ; 07/11/2014
608                                 mi_8:
609 00000498 0FB307                     btr     [edi], eax      ; clear bit 0 to bit x (1 to 31)
610                                     ;dec    bl
611 0000049B FEC9                       dec     cl
612 0000049D 7404                       jz      short mi_9
613 0000049F FEC0                       inc     al
614 000004A1 EBF5                       jmp     short mi_8
615                                 mi_9:
616                                     ;
617                                     ; Reset Kernel's Page Dir. and Page Table bits in M.A.T.
618                                     ;             (allocate pages for system page tables)
619
620                                     ; edx = MEM_ALLOC_TBL
621 000004A3 8B0D[AC800100]             mov     ecx, [memory_size] ; memory size in pages (PTEs)
622 000004A9 81C1FF030000               add     ecx, 1023       ; round up (1024 PTEs per table)
623 000004AF C1E90A                     shr     ecx, 10                 ; convert memory page count to
624                                                             ; page table count (PDE count)
625                                     ;
626 000004B2 51                         push    ecx             ; (**) PDE count (<= 1024)
627                                     ;
628 000004B3 41                         inc     ecx             ; +1 for kernel page directory
629                                     ;
630 000004B4 290D[B0800100]             sub     [free_pages], ecx ; 07/11/2014
631                                     ;
632 000004BA 8B35[A8800100]             mov     esi, [k_page_dir] ; Kernel's Page Directory address
633 000004C0 C1EE0C                     shr     esi, 12                 ; convert to page number
634                                 mi_10:
635 000004C3 89F0                       mov     eax, esi        ; allocation bit offset
636 000004C5 89C3                       mov     ebx, eax
637 000004C7 C1EB03                     shr     ebx, 3          ; convert to alloc. byte offset
638 000004CA 80E3FC                     and     bl,  0FCh       ; clear bit 0 and bit 1
639                                                             ;   to align on dword boundary
640 000004CD 83E01F                     and     eax, 31                 ; set allocation bit position
641                                                             ;  (bit 0 to bit 31)
642                                     ;
643 000004D0 01D3                       add     ebx, edx        ; offset in M.A.T. + M.A.T. address
644                                     ;
645 000004D2 0FB303                     btr     [ebx], eax      ; reset relevant bit (0 to 31)
646                                     ;
647 000004D5 46                         inc     esi             ; next page table
648 000004D6 E2EB                       loop    mi_10           ; allocate next kernel page table
649                                                             ; (ecx = page table count + 1)
650                                     ;
```

```
000004D8 59                             pop    ecx           ; (**) PDE count (= pg. tbl. count)
                                        ;
                                        ; Initialize Kernel Page Directory and Kernel Page Tables
                                        ;
                                        ; Initialize Kernel's Page Directory
000004D9 8B3D[A8800100]                 mov    edi, [k_page_dir]
000004DF 89F8                           mov    eax, edi
000004E1 0C03                           or     al, PDE_A_PRESENT + PDE_A_WRITE
                                                           ; supervisor + read&write + present
000004E3 89CA                           mov    edx, ecx     ; (**) PDE count (= pg. tbl. count)
                                mi_11:
000004E5 0500100000                     add    eax, 4096    ; Add page size (PGSZ)
                                                             ; EAX points to next page table
000004EA AB                             stosd
000004EB E2F8                           loop   mi_11
000004ED 29C0                           sub    eax, eax     ; Empty PDE
000004EF 66B90004                       mov    cx, 1024     ; Entry count (PGSZ/4)
000004F3 29D1                           sub    ecx, edx
000004F5 7402                           jz     short mi_12
000004F7 F3AB                           rep    stosd        ; clear remain (empty) PDEs
                                        ;
                                        ; Initialization of Kernel's Page Directory is OK, here.
                                mi_12:
                                        ; Initialize Kernel's Page Tables
                                        ;
                                        ; (EDI points to address of page table 0)
                                        ; eax = 0
000004F9 8B0D[AC800100]                 mov    ecx, [memory_size] ; memory size in pages
000004FF 89CA                           mov    edx, ecx     ; (***)
00000501 B003                           mov    al, PTE_A_PRESENT + PTE_A_WRITE
                                                          ; supervisor + read&write + present
                                mi_13:
00000503 AB                             stosd
00000504 0500100000                     add    eax, 4096
00000509 E2F8                           loop   mi_13
0000050B 6681E2FF03                     and    dx, 1023     ; (***)
00000510 740B                           jz     short mi_14
00000512 66B90004                       mov    cx, 1024
00000516 6629D1                         sub    cx, dx       ; from dx (<= 1023) to 1024
00000519 31C0                           xor    eax, eax
0000051B F3AB                           rep    stosd        ; clear remain (empty) PTEs
                                                            ; of the last page table
                                mi_14:
                                        ;  Initialization of Kernel's Page Tables is OK, here.
                                        ;
0000051D 89F8                           mov    eax, edi     ; end of the last page table page
                                                             ; (beginging of user space pages)
0000051F C1E80F                         shr    eax, 15            ; convert to M.A.T. byte offset
00000522 24FC                           and    al, 0FCh     ; clear bit 0 and bit 1 for
                                                            ; aligning on dword boundary
00000524 A3[BC800100]                   mov    [first_page], eax
00000529 A3[B4800100]                   mov    [next_page], eax ; The first free page pointer
                                                             ; for user programs
                                                             ; (Offset in Mem. Alloc. Tbl.)
                                        ;
                                        ; Linear/FLAT (1 to 1) memory paging for the kernel is OK, here.
                                        ;

                                        ; Enable paging
                                        ;
0000052E A1[A8800100]                   mov   eax, [k_page_dir]
00000533 0F22D8                         mov    cr3, eax
00000536 0F20C0                         mov    eax, cr0
00000539 0D00000080                     or     eax, 80000000h      ; set paging bit (bit 31)
0000053E 0F22C0                         mov    cr0, eax
                                        ;jmp    KCODE:StartPMP

00000541 EA                             db 0EAh             ; Opcode for far jump
00000542 [48050000]                     dd StartPMP         ; 32 bit offset
00000546 0800                           dw KCODE            ; kernel code segment descriptor
                                StartPMP:
                                        ; 06/11//2014
                                        ; Clear video page 0
                                        ;
                                        ; Temporary Code
                                        ;
00000548 B9E8030000                     mov    ecx, 80*25/2
0000054D BF00800B00                     mov    edi, 0B8000h
                                        ; 30/01/2016
                                        ;xor   eax, eax     ; black background, black fore color
00000552 B800070007                     mov    eax, 07000700h  ; black background, light gray fore color
00000557 F3AB                           rep    stosd

                                        ; 19/08/2014
                                        ; Kernel Base Address = 0
                                        ; It is mapped to (physically) 0 in the page table.
                                        ; So, here is exactly 'StartPMP' address.

                                        ; 29/01/2016 (TRDOS 386 = TRDOS v2.0)
00000559 BE[1A440100]                   mov    esi, starting_msg
                                        ;; 14/08/2015 (kernel version message will appear
                                        ;;            when protected mode and paging is enabled)
0000055E BF00800B00                     mov    edi, 0B8000h ; 27/08/2014

                                        ; 30/11/2020
                                        ; 14/11/2020 (TRDOS 386 v2.0.3)
                                        ;cmp   byte [vbe3], 3 ; 03h
                                        ;jne   short pkv_1
                                        ;;mov  ah, 0Bh ; Black background, light cyan forecolor
                                        ;; Light red TRDOS 386 version text shows VBE3 is ready !
                                        ;mov   ah, 0Ch ; Black background, light red forecolor
                                        ;jmp   short pkv_2
                                ;pkv_1:
```

```
00000563 B40A                          mov    ah, 0Ah ; Black background, light green forecolor
                                     ;pkv_2:
                                       ; 20/08/2014
00000565 E8DA030000                    call   printk

                                       ; 'UNIX v7/x86' source code by Robert Nordier (1999)
                                       ; // Set IRQ offsets
                                       ;
                                       ;  Linux (v0.12) source code by Linus Torvalds (1991)
                                       ;
                                                                      ;; ICW1
0000056A B011                          mov    al, 11h                  ; Initialization sequence
0000056C E620                          out    20h, al                  ;      8259A-1
                                       ; jmp  $+2
0000056E E6A0                          out    0A0h, al          ;      8259A-2
                                                                      ;; ICW2
00000570 B020                          mov    al, 20h                  ; Start of hardware ints (20h)
00000572 E621                          out    21h, al                  ;      for 8259A-1
                                       ; jmp  $+2
00000574 B028                          mov    al, 28h                  ; Start of hardware ints (28h)
00000576 E6A1                          out    0A1h, al          ;      for 8259A-2
                                                                      ;
00000578 B004                          mov    al, 04h                  ;; ICW3
0000057A E621                          out    21h, al                  ;      IRQ2 of 8259A-1 (master)
                                       ; jmp  $+2
0000057C B002                          mov    al, 02h                  ;      is 8259A-2 (slave)
0000057E E6A1                          out    0A1h, al          ;
                                                                      ;; ICW4
00000580 B001                          mov    al, 01h                  ;
00000582 E621                          out    21h, al                  ;      8086 mode, normal EOI
                                       ; jmp  $+2
00000584 E6A1                          out    0A1h, al          ;      for both chips.

                                       ;mov   al, 0FFh     ; mask off all interrupts for now
                                       ;out   21h, al
                                       ;; jmp $+2
                                       ;out   0A1h, al

                                       ; 02/04/2015
                                       ; 26/03/2015 System call (INT 30h) modification
                                       ;  DPL = 3 (Interrupt service routine can be called from user mode)
                                       ;
                                       ;; Linux (v0.12) source code by Linus Torvalds (1991)
                                       ;  setup_idt:
                                       ;
                                         ;; 16/02/2015
                                       ;;mov     dword [DISKETTE_INT], fdc_int ; IRQ 6 handler
                                       ; 21/08/2014 (timer_int)
00000586 BE[B8400100]                  mov    esi, ilist
0000058B 8D3D[C07D0100]                lea    edi, [idt]
                                       ; 26/03/2015
00000591 B930000000                    mov    ecx, 48            ; 48 hardware interrupts (INT 0 to INT 2Fh)
                                       ; 02/04/2015
00000596 BB00000800                    mov    ebx,  80000h
                                     rp_sidt1:
0000059B AD                            lodsd
0000059C 89C2                          mov    edx, eax
0000059E 66BA008E                      mov    dx, 8E00h
000005A2 6689C3                        mov    bx, ax
000005A5 89D8                          mov    eax, ebx     ; /* selector = 0x0008 = cs */
                                                                    ; /* interrupt gate - dpl=0, present */
000005A7 AB                            stosd  ; selector & offset bits 0-15
000005A8 89D0                          mov    eax, edx
000005AA AB                            stosd  ; attributes & offset bits 16-23
000005AB E2EE                          loop   rp_sidt1
                                       ; 15/04/2016
                                       ; TRDOS 386 (TRDOS v2.0) /// 32 sofware interrupts ///
                                       ;mov   cl, 16        ; 16 software interrupts (INT 30h to INT 3Fh)
000005AD B120                          mov    cl, 32      ; 32 software interrupts (INT 30h to INT 4Fh)
                                     rp_sidt2:
000005AF AD                            lodsd
000005B0 21C0                          and    eax, eax
000005B2 7413                          jz     short rp_sidt3
000005B4 89C2                          mov    edx, eax
000005B6 66BA00EE                      mov    dx, 0EE00h   ; P=1b/DPL=11b/01110b
000005BA 6689C3                        mov    bx, ax
000005BD 89D8                          mov    eax, ebx     ; selector & offset bits 0-15
000005BF AB                            stosd
000005C0 89D0                          mov    eax, edx
000005C2 AB                            stosd
000005C3 E2EA                          loop   rp_sidt2
000005C5 EB16                          jmp    short sidt_OK
                                     rp_sidt3:
000005C7 B8[6D0D0000]                  mov    eax, ignore_int
000005CC 89C2                          mov    edx, eax
000005CE 66BA00EE                      mov    dx, 0EE00h   ; P=1b/DPL=11b/01110b
000005D2 6689C3                        mov    bx, ax
000005D5 89D8                          mov    eax, ebx     ; selector & offset bits 0-15
                                     rp_sidt4:
000005D7 AB                            stosd
000005D8 92                            xchg   eax, edx
000005D9 AB                            stosd
000005DA 92                            xchg   edx, eax
000005DB E2FA                          loop   rp_sidt4
                                     sidt_OK:
000005DD 0F011D[5E680000]              lidt   [idtd]
                                       ;
                                       ; TSS descriptor setup ; 24/03/2015
000005E4 B8[40800100]                  mov    eax, task_state_segment
000005E9 66A3[0A680000]                mov    [gdt_tss0], ax
000005EF C1C010                        rol    eax, 16
000005F2 A2[0C680000]                  mov    [gdt_tss1], al
000005F7 8825[0F680000]                mov    [gdt_tss2], ah
000005FD 66C705[A6800100]68-           mov    word [tss.IOPB], tss_end - task_state_segment
00000605 00
```

```
                                        ;
                                        ; IO Map Base address (When this address points
                                        ; to end of the TSS, CPU does not use IO port
                                        ; permission bit map for RING 3 IO permissions,
                                        ; access to any IO ports in ring 3 will be forbidden.)
                                        ;
                                    ;mov   [tss.esp0], esp ; TSS offset 4
                                    ;mov   word [tss.ss0], KDATA ; TSS offset 8 (SS)
00000606 66B82800                   mov    ax, TSS  ; It is needed when an interrupt
                                                    ; occurs (or a system call -software INT- is requested)
                                                    ; while cpu running in ring 3 (in user mode).
                                                    ; (Kernel stack pointer and segment will be loaded
                                                    ; from offset 4 and 8 of the TSS, by the CPU.)
0000060A 0F00D8                     ltr    ax  ; Load task register
                                    ;
                                esp0_set0:
                                    ; 30/07/2015
0000060D 8B0D[AC800100]             mov    ecx, [memory_size] ; memory size in pages
00000613 C1E10C                     shl    ecx, 12 ; convert page count to byte count
00000616 81F900004000               cmp    ecx, CORE ; beginning of user's memory space (400000h)
                                                     ; (kernel mode virtual address)
0000061C 7605                       jna    short esp0_set1
                                    ;
                                    ; If available memory > CORE (end of the 1st 4 MB)
                                    ; set stack pointer to CORE
                                    ;(Because, PDE 0 is reserved for kernel space in user's page directory)
                                    ;(PDE 0 points to page table of the 1st 4 MB virtual address space)
0000061E B900004000                 mov    ecx, CORE
                                esp0_set1:
00000623 89CC                       mov    esp, ecx ; top of kernel stack (**tss.esp0**)
                                esp0_set_ok:
                                    ; 30/07/2015 (**tss.esp0**)
00000625 8925[44800100]             mov    [tss.esp0], esp
0000062B 66C705[48800100]10-          mov     word [tss.ss0], KDATA
00000633 00
                                ; 14/08/2015
                                ; 10/11/2014 (Retro UNIX 386 v1 - Erdogan Tan)
                                ;
                                ;cli   ; Disable interrupts (for CPU)
                                ;    (CPU will not handle hardware interrupts, except NMI!)
                                ;
00000634 30C0                   xor    al, al       ; Enable all hardware interrupts!
00000636 E621                   out    21h, al             ; (IBM PC-AT compatibility)
00000638 EB00                   jmp    $+2          ; (All conventional PC-AT hardware
0000063A E6A1                   out    0A1h, al     ;  interrupts will be in use.)
                                                    ; (Even if related hardware component
                                                    ;  does not exist!)
                                ; Enable NMI
0000063C B07F                   mov    al, 7Fh             ; Clear bit 7 to enable NMI (again)
0000063E E670                   out    70h, al
                                ; 23/02/2015
00000640 90                     nop
00000641 E471                   in     al, 71h             ; read in 71h just after writing out to 70h
                                                    ; for preventing unknown state (!?)
                                ;
                                ; Only a NMI can occur here... (Before a 'STI' instruction)
                                ;
                                ; 02/09/2014
00000643 6631DB                 xor    bx, bx
00000646 66BA0002               mov    dx, 0200h    ; Row 2, column 0  ; 07/03/2015
0000064A E8AB1C0000             call   _set_cpos    ; 24/01/2016

                                ; 14/11/2020 (TRDOS 386 v2.0.3)
                                ; Check VBE3 protected mode interface/feature(s)

                                ;cmp   byte [vbe3], 3 ; 03h
                                ;jne   short display_mem_info

                                ; 20/11/2020
0000064F 803D[4E090000]02       cmp    byte [vbe3], 2 ; 02h
00000656 770B                   ja     short vbe3_pmid_chk
                                ;jb    short display_mem_info
00000658 0F82CA010000                  jb     display_mem_info ; 02/12/2020
0000065E E9FE010000                    jmp    check_boch_plex86_vbe
                                vbe3_pmid_chk:
00000663 B9EA7F0000                    mov    ecx, 32768 - (20+2) ; 32766 - PMInfoBlockSize
00000668 BE02000C00                    mov    esi, 0C0002h ; 1st word of the video bios rom is 0AA55h

                                chk_pmi_sign:
                                       ;mov   eax, [esi]
                                       ;cmp   eax, 'PMID'
                                       ; 30/11/2020
                                       ;cmp   al, 'P'
                                       ;jne   short chk_pmi_sign_next
0000066D 813E504D4944                  cmp    dword [esi], 'PMID'
                                       ;je    short display_vbios_product_name
00000673 740E                          je     short verify_pmib_chksum ; 15/11/2020
                                ;chk_pmi_sign_next:
00000675 46                            inc    esi  ; inc si
00000676 E2F5                          loop   chk_pmi_sign

                                not_valid_pmib:
00000678 FE0D[4E090000]                dec    byte [vbe3] ; 2 = VBE2 compatible
                                                           ; (vbe3 feature is defective in this vbios)
                                       ;jmp   short display_mem_info
                                       ; 02/12/2020
0000067E E9A5010000                    jmp    display_mem_info

                                verify_pmib_chksum:
                                       ; 15/11/2020
00000683 31C0                          xor    eax, eax
                                       ;mov   ecx, eax
                                       ;mov   cl, 20
00000685 66B91400                      mov    cx, 20 ; 30/11/2020
00000689 56                            push   esi
```

```
                                pmib_sum_bytes:
0000068A AC                             lodsb
0000068B 00C4                           add     ah, al
0000068D E2FB                           loop    pmib_sum_bytes
0000068F 5E                             pop     esi
00000690 08E4                           or      ah, ah
00000692 75E4                           jnz     short not_valid_pmib ; AH must be 0
                                        ; 28/02/2021
                                        ; Set default (initial) truecolor bpp value to 32
                                        ; (for VBE3 video bios.. because vbe3 video bioses
                                        ;  use 32bpp -for truecolor modes- instead of 24bpp)
                                        ; (This setting may be changed via 'sysvideo' bx=0908h)
00000694 C605[6F7D0100]20               mov     byte [truecolor], 32 ; (RGB: 00RRGGBBh)
                                display_vbios_product_name: ; 14/11/2020
                                        ; ESI points to 'PMID' (0C0000h + 'PMID' offset)
                                        ; 15/11/2020
                                        ;mov   [pmid_addr], si     ; PMInfoBlock offset
                                        ;             ; (in VGA bios, 0C0000h + offset)
                                        ; 02/12/2020
                                        ;push  esi ; * pmid_addr
0000069B 89F7                           mov     edi, esi
                                        ;mov   esi, [VBE3INFOBLOCK+22] ; 097E00h + 16h
                                        ;             ; OemVendorNamePtr (seg16:off16)
0000069D 8B35067E0900                   mov     esi, [VBE3INFOBLOCK+6] ; 097E00h + 06h
                                                      ; OemStringPtr (seg16:off16)
000006A3 30C0                           xor     al, al  ; eax = 0
000006A5 6696                           xchg    ax, si ; ax = offset, si = 0
000006A7 C1EE0C                         shr     esi, 12 ; (to convert segment to base addr)
000006AA 6601C6                         add     si, ax  ; esi has an address < 1 MB limit
                                                        ; (OemVendorName is in VBE3INFOBLOCK)
                                                        ; Example:
                                                        ; TRDOS 386 v2.0.3 VESA VBE3 protected mode
                                                        ; interface development reference is ...
                                                        ; NVIDIA GeForce FX5500 VGA BIOS -C000h:029Ch-
                                                        ; Version 4.34.20.54.00 -C000h:02EDh-
                                                        ; ((OemString is 'NVIDIA'))
                                                        ; ((OemVendorName is 'NVIDIA Corporation'))
                                                        ; ((OemProductName is 'NV34 Board - p162-1nz))
                                        ;mov   ah, 0Eh ; Black background, yellow forecolor
                                        ; 30/11/2020
000006AD B40C                           mov     ah, 0Ch ; Black background, light red forecolor
000006AF E8263B0000                     call    print_kmsg
                                        ;mov   ah, 07h
000006B4 BE[517D0100]                   mov     esi, vesa_vbe3_bios_msg
                                        ;call  print_kmsg
000006B9 E8223B0000                     call    pkmsg_loop ; 30/11/2020
                                        ; 02/12/2020
                                        ;pop   edi ; * pmid_addr
                                        ; 30/11/2020
                                        ; 29/11/2020 - TRDOS 386 v2.0.3
                                struc PMInfo  ;  VESA VBE3 PMInfoBlock ('PMID' block)
00000000 ????????                .Signature: resb 4  ; db 'PMID' ; PM Info Block Signature
00000004 ????                    .EntryPoint:      resw 1 ; Offset of PM entry point within BIOS
00000006 ????                    .PMInitialize: resw 1    ; Offset of PM initialization entry point
00000008 ????                    .BIOSDataSel:     resw 1       ; Selector to BIOS data area emulation block
0000000A ????                    .A0000Sel:  resw 1 ; Selector to access A0000h physical mem
0000000C ????                    .B0000Sel:  resw 1  ; Selector to access B0000h physical mem
0000000E ????                    .B8000Sel:  resw 1 ; Selector to access B8000h physical mem
00000010 ????                    .CodeSegSel:      resw 1 ; Selector to access code segment as data
00000012 ??                      .InProtectMode: resb 1 ; Set to 1 when in protected mode
00000013 ??                      .Checksum:  resb 1 ; Checksum byte for structure
                                 .size:
                                endstruc
                                        ; 29/11/2020
                                vbe3pminit:
                                        ; 30/11/2020
                                        ;cmp   byte [vbe3], 3 ; is VESA VBE3 PMI ready ?
                                        ;jne   short di4
                                        ; Allocate 64KB contiguous (kernel) memory block
000006BE 31C0                           xor     eax, eax
000006C0 B900000100                     mov     ecx, 65536
000006C5 E8CD580000                     call    allocate_memory_block
                                        ;jc    short di4
000006CA 0F8251010000                   jc      di0 ; 30/11/2020
                                        ; of course this block must be in the 1st 16MB
                                        ; because vbe3 pmi segments will be 16 bit segments
                                        ; (80286 type segment descriptors in GDT)
000006D0 A3[08120300]                   mov     [vbe3bios_addr], eax
                                        ; set [pmid_addr] to the new location
000006D5 BE00000C00                     mov     esi, 0C0000h
                                        ; 30/11/2020
000006DA 29F7                           sub     edi, esi ; izolate offset
000006DC 01C7                           add     edi, eax ; new address
000006DE 893D[0C120300]                 mov     [pmid_addr], edi ; new 'PMID' location
```

```
1069
1070                                    ; Move VIDEO BIOS from 0C0000h to EAX
1071 000006E4 B900400000                mov    ecx, 65536/4
1072 000006E9 89C7                      mov    edi, eax ; 30/11/2020
1073 000006EB F3A5                      rep    movsd
1074
1075                                    ; 02/12/2020
1076                                    ; 30/11/2020
1077                                    ; set vbe3 segment selectors
1078
1079                                    ; VBE3CS (VESA VBE3 video bios code segment)
1080 000006ED BF[12680000]              mov    edi, _vbe3_CS+2 ; base address bits 0..15
1081 000006F2 66AB                      stosw  ; edi = _vbe3_CS+4
1082 000006F4 C1C810                    ror    eax, 16
1083 000006F7 8807                      mov    [edi], al ; base address, bits 16..23
1084
1085                                    ; VBE3DS ('CodeSegSel' in PMInfoBlock)
1086 000006F9 BF[3C680000]              mov    edi, _vbe3_DS+4 ; base addr bits 16..23
1087 000006FE 8807                      mov    [edi], al
1088 00000700 C1C010                    rol    eax, 16
1089 00000703 668947FE                  mov    [edi-2], ax ; base address, bits 0..15
1090
1091                                    ; VBE3BDS (BIOSDataSel in PMInfoBlock)
1092 00000707 BF[1A680000]              mov    edi, _vbe3_BDS+2 ; base addr bits 0..15
1093 0000070C B800700900                mov    eax, VBE3BIOSDATABLOCK ; 1536 bytes
1094 00000711 66AB                      stosw  ; edi = _vbe3_BDS+4
1095 00000713 C1E810                    shr    eax, 16
1096 00000716 8807                      mov    [edi], al ; base address, bits 16..23
1097
1098                                    ; VBE3SS (1024 bytes)
1099 00000718 BF[42680000]              mov    edi, _vbe3_SS+2 ; base addr bits 0..15
1100 0000071D B800600900                mov    eax, VBE3STACKADDR ; size = 1024 bytes
1101 00000722 66AB                      stosw  ; edi =  vbe3_SS+4
1102 00000724 C1E810                    shr    eax, 16
1103 00000727 8807                      mov    [edi], al ; base address, bits 16..23
1104
1105                                    ; stack pointer (esp) will be set to 1020
1106                                    ; (before VBE3 PMI call)
1107
1108                                    ; VBE3ES (max: 2048 bytes)
1109 00000729 BF[4A680000]              mov    edi, _vbe3_ES+2 ; base addr bits 0..15
1110 0000072E B800760900                mov    eax, VBE3SAVERESTOREBLOCK
1111 00000733 66AB                      stosw  ; edi = _vbe3_ES+4
1112 00000735 C1E810                    shr    eax, 16
1113 00000738 8807                      mov    [edi], al ; base address, bits 16..23
1114
1115                                    ;Note: low word of _VBE3_ES base address will be
1116                                    ;      set -again- by VBE3 PMI caller routine
1117
1118                                    ; 09/12/2020
1119                                    ;; set pmi32 (as VBE3 PMI is ready)
1120                                    ;inc   byte [pmi32] ; = 1
1121
1122                                    ; KCODE16 (set PMI far return segment)
1123 0000073A BF[52680000]              mov    edi, _16bit_CS+2 ; base addr bits 0..15
1124 0000073F B8[C8070000]              mov    eax, pminit_return_addr16
1125 00000744 66AB                      stosw  ; edi = _16bit_CS+4
1126 00000746 C1E810                    shr    eax, 16
1127 00000749 8807                      mov    [edi], al ; base address, bits 16..23
1128
1129                                    ; 30/11/2020
1130                                    ; clear mem from VBE3 BIOS data area emu block
1131                                    ; to end of vbe3 buffers
1132
1133                                    ; 01/12/2020
1134 0000074B BF00700900                mov    edi, VBE3BIOSDATABLOCK ; 97000h
1135                                    ;mov   cx, (VBE3INFOBLOCK-VBE3BIOSDATABLOCK)/4
1136                                    ;      ; ecx = 3584/4 double words
1137                                    ; 21/12/2020
1138 00000750 66B90003                  mov    cx, (VBE3MODEINFOBLOCK-VBE3BIOSDATABLOCK)/4
1139                                           ; ecx = 3072/4 double words
1140                                    ;xor   eax, eax
1141 00000754 30C0                      xor    al, al
1142 00000756 F3AB                      rep    stosd
1143
1144                                    ; Filling PMInfoBlock selector fields
1145 00000758 8B3D[0C120300]            mov    edi, [pmid_addr]
1146 0000075E 66C747083800              mov    word [edi+PMInfo.BIOSDataSel], VBE3BDS
1147 00000764 66C7470A4000              mov    word [edi+PMInfo.A0000Sel], VBE3A000
1148 0000076A 66C7470C4800              mov    word [edi+PMInfo.B0000Sel], VBE3B000
1149 00000770 66C7470E5000              mov    word [edi+PMInfo.B8000Sel], VBE3B800
1150 00000776 66C747105800              mov    word [edi+PMInfo.CodeSegSel], VBE3DS
1151 0000077C C6471201                  mov    byte [edi+PMInfo.InProtectMode], 1
1152
1153                                    ; Calculate and write checksum byte
1154 00000780 89FE                      mov    esi, edi
1155 00000782 B113                      mov    cl, PMInfo.size - 1
1156                                    ;xor   ah, ah
1157                                pmid_chksum:
1158 00000784 AC                        lodsb
1159 00000785 00C4                      add    ah, al
1160 00000787 E2FB                      loop   pmid_chksum
1161 00000789 F6DC                      neg    ah ; 1 -> 255, 255 -> 1
1162 0000078B 8826                      mov    byte [esi], ah  ; checksum
1163
1164                                    ; far call PM initialization
1165                                    ; (VBE3 video bios will return via 'retf')
1166
1167 0000078D 668B4706                  mov    ax, [edi+PMInfo.PMInitialize]
1168                                    ; 30/11/2020
1169 00000791 C1E010                    shl    eax, 16 ; save entry address in hw
1170                                    ; ax = 0
1171
1172                                    ; 02/12/2020
1173 00000794 68[EC070000]              push   pminit_ok ; normal, near return address
```

```
1174
1175                                    ; 30/11/2020
1176                                _VBE3PMI_fcall:
1177                                    ; ax = function, hw of eax = entry address
1178 00000799 9C                       pushf  ; save 32 bit flags
1179 0000079A 56                       push   esi ; *
1180 0000079B 55                       push   ebp ; **
1181
1182 0000079C 89E5                     mov    ebp, esp ; save 32 bit stack pointer
1183
1184 0000079E 89C6                     mov    esi, eax
1185
1186 000007A0 FA                       cli
1187
1188                                    ; Disable interrupts (clear interrupt flag)
1189                                    ; Reset Interrupt MASK Registers (Master&Slave)
1190 000007A1 B0FF                     mov    al, 0FFh     ; mask off all interrupts
1191 000007A3 E621                     out    21h, al            ; on master PIC (8259)
1192 000007A5 EB00                     jmp    $+2  ; (delay)
1193 000007A7 E6A1                     out    0A1h, al     ; on slave PIC (8259)
1194
1195                                    ; 02/12/2020
1196 000007A9 66B86000                 mov    ax, VBE3SS
1197 000007AD 8ED0                     mov    ss, ax
1198
1199 000007AF BCFC030000               mov    esp, 1020 ; 30/11/2020
1200
1201                                    ; 01/12/2020
1202                                    ;lss   esp, [stack16]
1203
1204 000007B4 C1E810                   shr    eax, 16      ; now, entry address is in lw
1205
1206                                    ; 30/11/2020 - 16 bit pm selector test (OK)
1207                                    ; (32 bit stack push/pop & retf with 32 bit code segment)
1208                                    ; (16 bit stack push/pop with 16 bit code segment)
1209
1210                                    ; return
1211                                    ;push  KCODE16
1212                                    ;push  0 ; 30/11/2020 (pminit_return_addr16)
1213
1214                                    ; 30/11/2020 (16 bit stack during retf from video bios)
1215 000007B7 C7042400007000             mov    dword [esp], KCODE16 << 16
1216                                                        ; ip = 0, cs = KCODE16
1217                                    ; 01/12/2020
1218                                    ;mov   dword [VBE3STACKADDR+1020], KCODE16*65536
1219
1220                                    ;mov   [jumpfar16], eax
1221
1222                                    ; 02/12/2020
1223                                    ; 30/11/2020 (32 bit stack during retf from kernel)
1224                                    ; far jump/call via retf
1225 000007BE 6A30                     push   VBE3CS ; VBE3 video bios's code segment
1226 000007C0 50                       push   eax ; PMInitialize or EntryPoint
1227
1228 000007C1 6689F0                   mov    ax, si ; restore function
1229
1230                                    ; 02/12/2020
1231 000007C4 31F6                     xor    esi, esi ; (not necessary, it is not used)
1232
1233 000007C6 CB                       retf   ; far return (to 16 bit code segment)
1234
1235                                    ; 01/12/2020
1236                                    ;db    0EAh  ; far jump to 16 bit code segment
1237                                ;jumpfar16:
1238                                    ;dd    0
1239                                    ;dw    VBE3CS
1240
1241                                ;stack16:
1242                                    ;dd    1020
1243                                    ;dw    VBE3SS
1244
1245 000007C7 90                       align 2
1246
1247                                pminit_return_addr16:
1248                                    ; 02/12/2020
1249                                    ; 30/11/2020
1250                                    ;;db   66h              ; Prefix for 32-bit
1251                                    ;db    0EAh             ; Opcode for far jump
1252                                    ;dd    pminit_return_addr32 ; 32 bit Offset
1253                                    ;dw    KCODE            ; 32 bit code segment
1254                                    ; 01/12/2020
1255 000007C8 EA[CF070000]0800         jmp    KCODE:pminit_return_addr32
1256
1257                                pminit_return_addr32:
1258                                    ; restore 32 bit kernel selectors and 32 bit stack addr
1259 000007CF BE10000000               mov    esi, KDATA
1260 000007D4 8EDE                     mov    ds, si
1261 000007D6 8EC6                     mov    es, si
1262 000007D8 8ED6                     mov    ss, si
1263 000007DA 89EC                     mov    esp, ebp  ; top of stack = iretd return addr
1264
1265 000007DC 5D                       pop    ebp ; **
1266 000007DD 5E                       pop    esi ; *
1267 000007DE 9D                       popf   ; restore 32 bit flags
1268
1269                                    ; enable interrupts
1270
1271 000007DF FA                       cli
1272
1273                                    ; 21/12/2020
1274 000007E0 50                       push   eax
1275
1276 000007E1 30C0                     xor    al, al    ; Enable all hardware interrupts!
1277 000007E3 E621                     out    21h, al         ; (IBM PC-AT compatibility)
1278 000007E5 EB00                     jmp    $+2      ; (All conventional PC-AT hardware
```

```
1279 000007E7 E6A1                          out    0A1h, al   ;  interrupts will be in use.)
1280                                                         ; (Even if related hardware component
1281                                                         ;  does not exist!)
1282 000007E9 58                            pop    eax
1283
1284 000007EA FB                            sti
1285
1286                                        ; top of stack = return address
1287                                        ; ('pminit_ok' for PMinit)
1288
1289 000007EB C3                            retn
1290
1291                                  pminit_ok:
1292                                        ; 03/12/2020
1293                                        ; (set [pmid_addr] to PMI entry point for next calls)
1294 000007EC 8305[0C120300]04              add    dword [pmid_addr], PMInfo.EntryPoint ; + 4
1295
1296                                        ; 17/01/2021
1297                                        ; copy EDID data from temporary location to final address
1298 000007F3 803D[6D420000]4F              cmp    byte [edid], 4Fh
1299 000007FA 7511                          jne    short vbe3h_chcl
1300 000007FC B920000000                    mov    ecx, 32 ; 128 bytes, 32 dwords
1301 00000801 BE007D0900                    mov    esi, VBE3EDIDINFOBLOCK ; 97D00h
1302 00000806 BF[84110300]                  mov    edi, edid_info
1303 0000080B F3A5                          rep    movsd
1304                                        ; 17/01/2021
1305                                  vbe3h_chcl:
1306                                        ; 16/01/2021
1307                                        ;; 06/12/2020
1308                                        ;; Save video mode 03h regs/dac/bios state
1309                                        ;
1310                                        ;mov   ax, 4F04h ; VESA VBE Function 04h
1311                                        ;              ; Save/Restore State
1312                                        ;sub   dl, dl ; 0 = return buffer size
1313                                        ;mov   cx, 0Fh  ; ctrl/bios/dac/regs
1314                                        ;
1315                                        ;call  int10h_32bit_pmi
1316                                        ;; bx = number of 64-byte blocks to hold the state buff
1317                                        ;
1318                                        ;;mov  [vbe3stbufsize], bx
1319                                        ;; 16/01/2021
1320                                        ;or    word [vbe3stbsflags], 32768  ; set bit 15
1321                                        ;mov   ax, bx
1322                                        ;shl   ax, 6  ; * 64
1323                                        ;mov   [vbestatebufsize+30], ax
1324                                        ;
1325                                        ;; 06/12/2020
1326                                        ;; check 'vbe3stbufsize' (it must be <= 32)
1327                                        ;
1328                                        ;cmp   bx, 32
1329                                        ;ja    short display_mem_info ; light red forecolor
1330                                        ;
1331                                        ;; 16/01/2021
1332                                        ;or    byte [vbe3stbsflags], 1 ; set bit 0
1333
1334                                        ; 30/11/2020
1335                                        ; Change VESA VBE3 BIOS text color in order to give
1336                                        ; "VBE3 PMI initialization has been successed" meaning
1337
1338 0000080D BE40810B00                        mov    esi, 0B8000h + 160*2 ; row 2
1339 00000812 89F7                          mov    edi, esi
1340                                  vbe3h_chcl_next:
1341 00000814 66AD                          lodsw
1342 00000816 80FC0C                        cmp    ah, 0Ch      ; light red forecolor
1343 00000819 750D                          jne    short display_mem_info
1344 0000081B B40E                          mov    ah, 0Eh ; yellow forecolor
1345 0000081D 66AB                          stosw
1346 0000081F EBF3                          jmp    short vbe3h_chcl_next
1347
1348                                  di0:
1349                                        ; 30/11/2020
1350                                        ; Memory allocation error !
1351 00000821 C605[4E090000]00              mov    byte [vbe3], 0 ; disable VBE3
1352
1353                                  display_mem_info:
1354                                        ; 19/12/2020
1355                                        ; temporary
1356 00000828 E8C9390000                    call   default_lfb_info
1357                                        ;
1358                                        ; 06/11/2014
1359 0000082D E84F390000                    call   memory_info
1360                                        ; 14/08/2015
1361                                        ;call getch ; 28/02/2015
1362                                  drv_init:
1363 00000832 FB                            sti    ; Enable Interrupts
1364                                        ; 06/02/2015
1365 00000833 8B15[F0680000]                mov    edx, [hd0_type] ; hd0, hd1, hd2, hd3
1366 00000839 668B1D[EE680000]              mov    bx, [fd0_type] ; fd0, fd1
1367                                        ; 22/02/2015
1368 00000840 6621DB                        and    bx, bx
1369 00000843 756C                          jnz    short di1
1370                                        ;
1371 00000845 09D2                          or     edx, edx
1372 00000847 757A                          jnz    short di2
1373                                        ;
1374                                  setup_error:
1375 00000849 BE[BE430100]                  mov    esi, setup_error_msg
1376                                  psem:
1377 0000084E AC                            lodsb
1378 0000084F 08C0                          or     al, al
1379                                        ;jz    short haltx ; 22/02/2015
1380 00000851 747B                          jz     short di3
1381 00000853 56                            push   esi
1382                                        ; 13/05/2016
1383 00000854 BB07000000                    mov    ebx, 7 ; Black background,
```

```
1384                                                    ; light gray forecolor
1385                                                    ; Video page 0 (BH=0)
1386 00000859 E8051A0000                        call    _write_tty
1387 0000085E 5E                                pop     esi
1388 0000085F EBED                              jmp     short psem
1389
1390                                    check_boch_plex86_vbe:
1391                                            ; 20/11/2020
1392                                            ; check Bochs/Plex86 VGABios VBE extension
1393                                            ; (check if TRDOS 386 v2 is running on emulators)
1394                                            ; BOCHS/QEMU/VIRTUALBOX
1395                                            ;
1396                                            ; ref: vbe_display_api.txt
1397
1398                                            ; bochs/plex86 VGAbios VBE source code
1399                                            ;  by Jeroen Janssen (2002)
1400                                            ;  and Volker Ruppert (2003-2020)
1401
1402 00000861 29C0                             sub     eax, eax ; 0
1403 00000863 66BACE01                         mov     dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
1404 00000867 66EF                             out     dx, ax ; VBE_DISPI_INDEX_ID register
1405                                            ;mov    ax, 0B0C0h ; VBE_DISPI_ID0
1406                                            ;mov    dx, 1CFh ; VBE_DISPI_IOPORT_DATA
1407 00000869 6642                             inc     dx
1408                                            ;out    dx, ax
1409                                            ;nop
1410 0000086B 66ED                             in      ax, dx
1411 0000086D 80FCB0                           cmp     ah, 0B0h
1412                                            ;jne    short not_boch_qemu_vbe
1413 00000870 75B6                             jne     short display_mem_info
1414 00000872 3CC5                             cmp     al, 0C5h ; it must be 0B0C4h or 0B0C5h ..
1415                                            ;ja     short not_boch_qemu_vbe
1416 00000874 77B2                             ja      short display_mem_info
1417 00000876 3CC0                             cmp     al, 0C0h ; 0B0C0h to 0B0C5h .. ; Qemu
1418                                            ;cmp    al, 0C4h ; 0BC04h or 0B0C5h is OK ; Bochs
1419                                            ;jb     short not_boch_qemu_vbe
1420 00000878 72AE                             jb      short display_mem_info
1421
1422                                            ; save VESA VBE2 bios (bochs/qemu) signature
1423                                            ; for enabling VBE2 functions in TRDOS 386 v2 kernel
1424 0000087A A2[4F090000]                     mov     [vbe2bios], al ; 0C4h or 0C5h (for BOCHS)
1425                                                                   ; (0C0h-0C5h for QEMU)
1426 0000087F C605[5A7D0100]32                 mov     byte [vbe_vnumber], "2"
1427                                            ; 26/11/2020
1428                                            ; "BOCHS/QEMU/VIRTUALBOX VBE2 Video BIOS ..".
1429 00000886 BE[3C7D0100]                     mov     esi, vbe2_bochs_vbios ; BOCH/QEMU vbios msg
1430 0000088B B40E                             mov     ah, 0Eh  ; Yellow font
1431 0000088D E848390000                       call    print_kmsg
1432
1433                                            ; this is not necessary ! (20/11/2020)
1434 00000892 803D[4F090000]C4                 cmp     byte [vbe2bios], 0C4h
1435 00000899 728D                             jb      display_mem_info    ; (QEMU)
1436
1437                                            ; Display kernel version message if 0E9h hack port
1438                                            ; is enabled (bochs emulator feature)
1439 0000089B 66BAE900                         mov     dx, 0E9h ; hack port for BOCHS
1440 0000089F BE[957D0100]                     mov     esi, kernel_version_msg
1441                                    kvmsg_next_char:
1442 000008A4 AC                               lodsb
1443 000008A5 08C0                             or      al, al
1444 000008A7 7505                             jnz     short put_kvmsg_in_hack_port
1445                                    not_boch_qemu_vbe:
1446 000008A9 E97AFFFFFF                       jmp     display_mem_info
1447                                    put_kvmsg_in_hack_port:
1448 000008AE EE                               out     dx, al
1449 000008AF EBF3                             jmp     short kvmsg_next_char
1450
1451                                    di1:
1452                                            ; supress 'jmp short T6'
1453                                            ;  (activate fdc motor control code)
1454 000008B1 66C705[B0090000]90-              mov     word [T5], 9090h ; nop
1454 000008B9 90
1455                                            ;
1456                                            ;mov   ax, int_0Eh  ; IRQ 6 handler
1457                                            ;mov   di, 0Eh*4    ; IRQ 6 vector
1458                                            ;stosw
1459                                            ;mov   ax, cs
1460                                            ;stosw
1461                                            ;; 16/02/2015
1462                                              ;;mov     dword [DISKETTE_INT], fdc_int ; IRQ 6 handler
1463                                            ;
1464 000008BA E825480000                       CALL   DSKETTE_SETUP; Initialize Floppy Disks
1465                                            ;
1466 000008BF 09D2                             or      edx, edx
1467 000008C1 740B                             jz      short di3
1468                                    di2:
1469 000008C3 E85A480000                       call   DISK_SETUP    ; Initialize Fixed Disks
1470 000008C8 0F827BFFFFFF                     jc      setup_error
1471                                    di3:
1472 000008CE E882380000                       call    setup_rtc_int; 22/05/2015 (dsectrpm.s)
1473                                            ;
1474 000008D3 E87C390100                       call    display_disks ; 07/03/2015  (Temporary)
1475                                    ;haltx:
1476                                            ; 14/08/2015
1477                                            ;call  getch ; 22/02/2015
1478                                            ;sti   ; Enable interrupts (for CPU)
1479                                    ;       ; 29/01/2016
1480                                    ;       sub     ah, ah ;  read time count
1481                                    ;       call    int1Ah
1482                                    ;       mov     edx, ecx ; 18.2 * seconds
1483                                    ;md_info_msg_wait1:
1484                                    ;       ; 29/01/2016
1485                                    ;       mov     ah, 1
1486                                    ;       call    int16h
1487                                    ;       jz      short md_info_msg_wait2
```

```
1488                                ;     xor   ah, ah ; 0
1489                                ;     call  int16h
1490                                ;     jmp   short md_info_msg_ok
1491                                ;md_info_msg_wait2:
1492                                ;     sub   ah, ah  ; read time count
1493                                ;     call  int1Ah
1494                                ;     cmp   edx, ecx ; ; 18.2 * seconds
1495                                ;     jna   short md_info_msg_wait3
1496                                ;     xchg  edx, ecx
1497                                ;md_info_msg_wait3:
1498                                ;     sub   ecx, edx
1499                                ;     cmp   ecx, 127 ; 7 seconds (18.2 * 7)
1500                                ;     jb    short md_info_msg_wait1
1501                                ;md_info_msg_ok:
1502
1503                                      ; 15/12/2020
1504                                      ; set initial values of LFB parameters
1505
1506 000008D8 803D[4E090000]02            cmp   byte [vbe3], 2
1507 000008DF 7214                     jb    short di4
1508
1509                                      ;mov  ax, [def_LFB_addr]
1510                                      ;shl  eax, 16
1511                                      ;mov  [LFB_ADDR], eax
1512                                      ;mov  eax, 1024*768*3
1513                                      ;mov  [LFB_SIZE], eax
1514
1515 000008E1 BEFE7B0900                  mov   esi, VBE3MODEINFOBLOCK - 2
1516 000008E6 66C7061801                  mov   word [esi], 0118h ; default vbe mode
1517                                                               ; 1024*768, 24bpp
1518 000008EB E8F1300000                  call  set_lfbinfo_table
1519
1520 000008F0 E8DB5B0000                  call  allocate_lfb_pages_for_kernel
1521                                 di4:
1522                                      ; 08/09/2016
1523 000008F5 0F20C0                      mov   eax, cr0
1524 000008F8 A810                        test  al, 10h  ; Bit 4, ET (Extension Type)
1525 000008FA 7408                        jz    short sysinit
1526                                      ; 27/02/2017
1527 000008FC FE05[648D0100]              inc   byte [fpready]
1528                                      ; 80387 (FPU) is ready
1529 00000902 DBE3                        fninit ; Initialize Floating-Point Unit
1530                                 sysinit:
1531                                      ; 30/06/2015
1532 00000904 E81D660000                  call  sys_init
1533                                      ;
1534                                      ;jmp  cpu_reset ; 22/02/2015
1535                                 hang:
1536                                      ; 23/02/2015
1537                                      ;sti               ; Enable interrupts
1538 00000909 F4                          hlt
1539                                      ;
1540                                      ;nop
1541                                      ;; 03/12/2014
1542                                      ;; 28/08/2014
1543                                      ;mov  ah, 11h
1544                                      ;call getc
1545                                      ;jz     _c8
1546                                      ;
1547                                      ; 23/02/2015
1548                                      ; 06/02/2015
1549                                      ; 07/09/2014
1550 0000090A 31DB                        xor   ebx, ebx
1551 0000090C 8A1D[D6800100]              mov   bl, [ptty]   ; active_page
1552 00000912 89DE                        mov   esi, ebx
1553 00000914 66D1E6                      shl   si, 1
1554 00000917 81C6[D8800100]              add   esi, ttychr
1555 0000091D 668B06                      mov   ax, [esi]
1556 00000920 6621C0                      and   ax, ax
1557                                      ;jz   short _c8
1558 00000923 74E4                        jz    short hang
1559 00000925 66C7060000                  mov   word [esi], 0
1560 0000092A 80FB03                      cmp   bl, 3        ; Video page 3
1561                                      ;jb   short _c8
1562 0000092D 72DA                        jb    short hang
1563                                      ;
1564                                      ; 13/05/2016
1565                                      ; 07/09/2014
1566                                 nxtl:
1567                                      ;push bx
1568                                      ; 18/04/2021
1569 0000092F 53                          push  ebx
1570 00000930 66BB0E00                    mov   bx, 0Eh      ; Yellow character
1571                                                         ; on black background
1572                                                         ; bh = 0 (video page 0)
1573                                                         ; Retro UNIX 386 v1 - Video Mode 0
1574                                                         ; (PC/AT Video Mode 3 - 80x25 Alpha.)
1575                                      ;push ax
1576                                      ; 18/04/2021
1577 00000934 50                          push  eax
1578 00000935 E829190000                  call  _write_tty
1579                                      ;pop  ax
1580                                      ; 18/04/2021
1581 0000093A 58                          pop   eax
1582                                      ;pop  bx
1583 0000093B 5B                          pop   ebx
1584 0000093C 3C0D                        cmp   al, 0Dh            ; carriage return (enter)
1585                                      ;jne  short _c8
1586 0000093E 75C9                        jne   short hang
1587 00000940 B00A                        mov   al, 0Ah            ; next line
1588 00000942 EBEB                        jmp   short nxtl
1589
1590                                 ;_c8:
1591                                 ;    ; 25/08/2014
1592                                 ;    cli                      ; Disable interrupts
```

```
1593                                    ;       mov     al, [scounter + 1]
1594                                    ;       and     al, al
1595                                    ;       jnz     hang
1596                                    ;       call    rtc_p
1597                                    ;       jmp      hang
1598
1599                                            ; 27/08/2014
1600                                            ; 20/08/2014
1601                                    printk:
1602                                              ;mov    edi, [scr_row]
1603                                    pkl:
1604 00000944 AC                               lodsb
1605 00000945 08C0                             or      al, al
1606 00000947 7404                             jz      short pkr
1607 00000949 66AB                             stosw
1608 0000094B EBF7                             jmp     short pkl
1609                                    pkr:
1610 0000094D C3                               retn
1611
1612                                    ; 14/11/2020 (TRDOS 386 v2.0.3)
1613 0000094E 00                       vbe3: db 0  ; VESA VBE version (must be 03h)
1614                                          ; for using video bios calls in protected mode
1615                                    vbe2bios:
1616 0000094F B0                               db 0B0h ;
1617                                    ;pmid_addr:
1618                                            ;dw 0  ; > 0 if 'PMID' sign is found
1619                                            ;       ;       ('pmid' offset addr in VGA bios seg, 0C000h)
1620                                            ;; 02/12/2020
1621                                            ;dw 0  ; 32 bit address in pmid_addr
1622
1623                                    ; 28/02/2017
1624                                    ; 22/01/2017
1625                                    ; 15/01/2017
1626                                    ; 14/01/2017
1627                                    ; 02/01/2017
1628                                    ; 25/12/2016
1629                                    ; 19/12/2016
1630                                    ; 10/12/2016 (callback)
1631                                    ; 06/06/2016
1632                                    ; 23/05/2016
1633                                    ; 22/05/2016 - TRDOS 386 (TRDOS v2.0) Timer Event Modifications
1634                                    ; 25/07/2015
1635                                    ; 14/05/2015 (multi tasking -time sharing- 'clock', x_timer)
1636                                    ; 17/02/2015
1637                                    ; 06/02/2015 (unix386.s)
1638                                    ; 11/12/2014 - 22/12/2014 (dsectrm2.s)
1639                                    ;
1640                                    ; IBM PC-XT Model 286 Source Code - BIOS2.ASM (06/10/85)
1641                                    ;
1642                                    ;-- HARDWARE INT  08 H - ( IRQ LEVEL 0 ) ----------------------------------------
1643                                    ;       THIS ROUTINE HANDLES THE TIMER INTERRUPT FROM FROM CHANNEL 0 OF          :
1644                                    ;       THE 8254 TIMER.  INPUT FREQUENCY IS 1.19318 MHZ AND THE DIVISOR         :
1645                                    ;       IS 65536, RESULTING IN APPROXIMATELY 18.2 INTERRUPTS EVERY SECOND.      :
1646                                    ;                                                                               :
1647                                    ;       THE INTERRUPT HANDLER MAINTAINS A COUNT (40:6C) OF INTERRUPTS SINCE      :
1648                                    ;       POWER ON TIME, WHICH MAY BE USED TO ESTABLISH TIME OF DAY.               :
1649                                    ;       THE INTERRUPT HANDLER ALSO DECREMENTS THE MOTOR CONTROL COUNT (40:40)    :
1650                                    ;       OF THE DISKETTE, AND WHEN IT EXPIRES, WILL TURN OFF THE                  :
1651                                    ;       DISKETTE MOTOR(s), AND RESET THE MOTOR RUNNING FLAGS.                    :
1652                                    ;       THE INTERRUPT HANDLER WILL ALSO INVOKE A USER ROUTINE THROUGH            :
1653                                    ;       INTERRUPT 1CH AT EVERY TIME TICK.  THE USER MUST CODE A                  :
1654                                    ;       ROUTINE AND PLACE THE CORRECT ADDRESS IN THE VECTOR TABLE.               :
1655                                    ;-------------------------------------------------------------------------------
1656                                    ;
1657
1658                                    timer_int:  ; IRQ 0
1659                                    ;int_08h:   ; Timer
1660                                            ; 14/10/2015
1661                                            ; Here, we are simulating system call entry (for task switch)
1662                                            ; (If multitasking is enabled,
1663                                            ; 'clock' procedure may jump to 'sysrelease')
1664
1665 00000950 1E                               push  ds
1666 00000951 06                               push  es
1667 00000952 0FA0                             push  fs
1668 00000954 0FA8                             push  gs
1669
1670 00000956 60                               pushad ; eax, ecx, edx, ebx, esp -before pushad-, ebp, esi, edi
1671 00000957 66B91000                         mov   cx, KDATA
1672 0000095B 8ED9                             mov   ds, cx
1673 0000095D 8EC1                             mov   es, cx
1674 0000095F 8EE1                             mov   fs, cx
1675 00000961 8EE9                             mov   gs, cx
1676
1677 00000963 0F20D9                           mov    ecx, cr3
1678 00000966 890D[5C040300]                   mov    [cr3reg], ecx ; save current cr3 register value/content
1679
1680                                           ; 14/01/2017
1681 0000096C 3B0D[A8800100]                   cmp    ecx, [k_page_dir]
1682 00000972 7409                             je     short T3
1683
1684 00000974 8B0D[A8800100]                   mov    ecx, [k_page_dir]
1685 0000097A 0F22D9                           mov    cr3, ecx
1686                                    T3:
1687                                           ;sti                        ; INTERRUPTS BACK ON
1688 0000097D 66FF05[28810100]                 INC    word [TIMER_LOW]    ; INCREMENT TIME
1689 00000984 7507                             JNZ    short T4            ; GO TO TEST_DAY
1690 00000986 66FF05[2A810100]                 INC    word [TIMER_HIGH]   ; INCREMENT HIGH WORD OF TIME
1691                                    T4:                                ; TEST_DAY
1692 0000098D 66833D[2A810100]18               CMP    word [TIMER_HIGH],018H    ; TEST FOR COUNT EQUALING 24 HOURS
1693 00000995 7519                             JNZ    short T5            ; GO TO DISKETTE_CTL
1694 00000997 66813D[28810100]B0-              CMP    word [TIMER_LOW],0B0H
1694 0000099F 00
1695 000009A0 750E                             JNZ    short T5            ; GO TO DISKETTE_CTL
1696
```

```
                                ;-----      TIMER HAS GONE 24 HOURS
                                      ;;SUB  AX,AX
                                      ;MOV   [TIMER_HIGH],AX
                                      ;MOV   [TIMER_LOW],AX
000009A2 29C0                         sub    eax, eax
000009A4 A3[28810100]                 mov    [TIMER_LH], eax
                                      ;
000009A9 C605[2C810100]01             MOV    byte [TIMER_OFL],1

                                ;-----      TEST FOR DISKETTE TIME OUT

                                T5:
                                      ; 23/12/2014
000009B0 EB1D                         jmp    short T6           ; will be replaced with nop, nop
                                                                ; (9090h) if a floppy disk
                                                                ; is detected.
                                      ;mov   al,[CS:MOTOR_COUNT]
000009B2 A0[2F810100]                 mov    al, [MOTOR_COUNT]
000009B7 FEC8                         dec    al
                                      ;mov   [CS:MOTOR_COUNT], al      ; DECREMENT DISKETTE MOTOR CONTROL
000009B9 A2[2F810100]                 mov    [MOTOR_COUNT], al
                                      ;mov   [ORG_MOTOR_COUNT], al
000009BE 750F                         JNZ    short T6           ; RETURN IF COUNT NOT OUT
000009C0 B0F0                         mov    al,0F0h
                                      ;AND   [CS:MOTOR_STATUS],al      ; TURN OFF MOTOR RUNNING BITS
000009C2 2005[2E810100]               and    [MOTOR_STATUS], al
                                      ;and   [ORG_MOTOR_STATUS], al
000009C8 B00C                         MOV    AL,0CH             ; bit 3 = enable IRQ & DMA,
                                                                ; bit 2 = enable controller
                                                                ;      1 = normal operation
                                                                ;      0 = reset
                                                                ; bit 0, 1 = drive select
                                                                ; bit 4-7 = motor running bits
000009CA 66BAF203                     MOV    DX,03F2H           ; FDC CTL PORT
000009CE EE                           OUT    DX,AL              ; TURN OFF THE MOTOR
                                T6:
                                      ;inc   word [CS:wait_count]      ; 22/12/2014 (byte -> word)
                                                                ; TIMER TICK INTERRUPT
                                      ;;inc  word [wait_count] ;;27/02/2015
                                      ;INT   1CH                ; TRANSFER CONTROL TO A USER ROUTINE
                                      ;cli
000009CF E855040000                   call   u_timer                   ; TRANSFER CONTROL TO A USER ROUTINE
                                      ; 23/05/2016
000009D4 E8211D0100                   call   clock              ; Multi Tasking control procedure
                                T7:
                                      ; 14/10/2015
000009D9 B020                         MOV    AL,EOI             ; GET END OF INTERRUPT MASK
000009DB FA                           CLI                       ; DISABLE INTERRUPTS TILL STACK CLEARED
000009DC E620                         OUT    INTA00,AL          ; END OF INTERRUPT TO 8259 - 1
                                      ;
                                rtc_int_2:
                                      ; 26/12/2016
                                      ;mov   ecx, [cr3reg]
                                      ; 13/01/2017
000009DE 803D[D4030300]00             cmp    byte [u.t_lock], 0        ; T_LOCK
000009E5 7730                         ja     short timer_int_return  ; Timer Lock : 'sysrele' is needed !
                                      ; 28/02/2017
                                      ; We need to exit if the user's IRQ callback service is in progress!
                                      ; (To prevent a conflict!)
000009E7 803D[D8030300]00             cmp    byte [u.r_lock], 0  ; R_LOCK, IRQ callback service lock !
000009EE 7727                         ja     short timer_int_return  ; Timer Lock : 'sysrele' is needed !
                                      ; 15/01/2017
000009F0 803D[388D0100]02             cmp    byte [priority], 2
000009F7 733A                         jnb    short T8  ; current process has a timer event (15/01/2017)
                                      ; 22/05/2016
000009F9 803D[398D0100]00             cmp    byte [p_change], 0 ; in 'set_run_sequence', in 'rtc_p'
00000A00 7615                         jna    short timer_int_return ; 23/05/2016

                                      ; 15/01/2017

                                      ; present process must be changed with high priority process
                                      ;xor   al, al
00000A02 31C0                         xor    eax, eax ; 26/12/2016
00000A04 A2[398D0100]                 mov    [p_change], al ; 0
                                      ;mov   byte [priority], 2 ; 15/01/2017 (there is a timer event)

00000A09 803D[5B030300]FF             cmp    byte [sysflg], 0FFh ; user or system space ?
00000A10 7416                         je     short rtc_int_3     ; user space ([sysflg]= 0FFh)

                                      ; system space, wait for 'sysret'
                                      ; to change running process
                                      ; with high priority (event) process

00000A12 A2[A8030300]                 mov    [u.quant], al ; 0

                                timer_int_return: ; 23/05/2016 - jump from 'rtc_int' ('rtc_int_2')
00000A17 8B0D[5C040300]               mov    ecx, [cr3reg]       ; previous value/content of cr3 register
00000A1D 0F22D9                       mov    cr3, ecx     ; restore cr3 register content
                                      ;
00000A20 61                           popad ; edi, esi, ebp, temp (icrement esp by 4), ebx, edx, ecx, eax
                                      ;
00000A21 0FA9                         pop    gs
00000A23 0FA1                         pop    fs
00000A25 07                           pop    es
00000A26 1F                           pop    ds
                                      ;
00000A27 CF                           iretd  ; return from interrupt

                                rtc_int_3:
00000A28 FE05[5B030300]               inc    byte [sysflg]       ; now, we are in system space
                                      ;
00000A2E E9BDC90000                   jmp   sysrelease ; change running process immediatelly

                                T8:
                                      ; 13/01/2017 (eax -> ebx)
```

```
                                        ; callback checking... (19/12/2016)
00000A33 31DB                           xor    ebx, ebx
00000A35 871D[D0030300]                 xchg   ebx, [u.tcb] ; callback address (0 = normal return)
00000A3B 09DB                           or     ebx, ebx
00000A3D 74D8                           jz     short timer_int_return
                                        
                                        ; Set user's callback routine as return address from this interrupt
                                        ; and set normal return address as return address from callback
                                        ; routine!!! (19/12/2016)
                                        
                                        ; 14/01/2017
                                        ; 13/01/2017 - Timer Lock (T_LOCK)
00000A3F FE05[D4030300]                 inc    byte [u.t_lock]
00000A45 8A0D[5B030300]                 mov    cl, [sysflg]
00000A4B 880D[D5030300]                 mov    [u.t_mode], cl
                                        
00000A51 8B2D[44800100]                 mov    ebp, [tss.esp0] ; kernel stack address (for ring 0)
00000A57 83ED14                         sub    ebp, 20             ; eip, cs, eflags, esp, ss
00000A5A 892D[5C030300]                 mov    [u.sp], ebp
00000A60 8925[60030300]                 mov    [u.usp], esp
                                        
                                        ;or    word [ebp+8], 200h ; 22/01/2017, force enabling interrupts
                                        
00000A66 8B44241C                       mov    eax, [esp+28] ; pushed eax
00000A6A A3[64030300]                   mov    [u.r0], eax
                                        
00000A6F E882090100                     call   wswap ; save user's registers & status
                                        
                                        ; software int is in ring 0 but timer int must return to ring 3
                                        ; so, ring 3 return address and stack registers
                                        ; (eip, cs, eflags, esp, ss)
                                        ; must be copied to timer int return
                                        ; eip will be replaced by callback service routine address
                                        
00000A74 C605[5B030300]FF               mov    byte [sysflg], 0FFh ; user mode
                                        
                                        ; system mode (system call)
                                        ;mov   ebp, [u.sp] ; EIP (u), CS (UCODE), EFLAGS (u),
                                                           ; ESP (u), SS (UDATA)
                                        
00000A7B 8B4510                         mov    eax, [ebp+16]; SS (UDATA
00000A7E 89E6                           mov    esi, esp
00000A80 50                             push   eax
00000A81 50                             push   eax
00000A82 89E7                           mov    edi, esp
00000A84 893D[60030300]                 mov    [u.usp], edi
00000A8A B908000000                     mov    ecx, ((ESPACE/4) - 4) ; except DS, ES, FS, GS
00000A8F F3A5                           rep    movsd
00000A91 B104                           mov    cl, 4
00000A93 F3AB                           rep    stosd
00000A95 893D[5C030300]                 mov    [u.sp], edi
00000A9B 89EE                           mov    esi, ebp
00000A9D B105                           mov    cl, 5 ; EIP (u), CS (UCODE), EFLAGS (u), ESP (u), SS (UDATA)
00000A9F F3A5                           rep    movsd
                                        
00000AA1 8B0D[B8030300]                 mov    ecx, [u.pgdir]
00000AA7 890D[5C040300]                 mov    [cr3reg], ecx
                                        
                                        ; 13/01/207 (eax -> ebx)
                                        ; EBX = callback routine address (virtual, not physical address!)
                                        
                                        ; 09/01/2017
                                        ; !!! CALLBACK ROUTINE MUST BE ENDED/RETURNED WITH 'sysrele'
                                        ;     system call !!!
                                        ; 25/12/2016
                                        ; Callback Note: (19/12/2016)
                                        ; !!! CALLBACK ROUTINE MUST BE ENDED/RETURNED WITH 'RETN' !!!
                                        ;     pushf ; save flags
                                        ;     <callback service code>
                                        ;     popf  ; restore flags
                                        ;     retn ; return to normal running address
                                        ;
                                        
                                        ; 15/01/2017
                                        ; 14/01/2017
                                        ; 13/01/2017 (eax -> ebx)
                                        ; 10/01/2017
                                    set_callback_addr:
                                        ; 09/01/2017 (**)
                                        ; 02/01/2017 (*)
                                        ; 25/12/2016 (*)
                                        ; 19/12/2016 (TRDOS 386 feature only!)
                                        ;
                                        ; This routine sets return address
                                        ; to start of user's interrupt
                                        ; service (callback) address
                                        ;; and sets callback 'retn' address to normal
                                        ;; return address of user's running code!
                                        ;
                                        ; INPUT:
                                        ;     EBX = callback routine/service address
                                        ;           (virtual, not physical address!)
                                        ;     [u.sp] = kernel stack, points to
                                        ;              user's EIP,CS,EFLAGS,ESP,SS
                                        ;              registers.
                                        ; OUTPUT:
                                        ;     EIP (user) = callback (service) address
                                        ;     CS (user) = UCODE
                                        ;     EFLAGS (user) = flags before callback
                                        ;      ESP (user) = ESP-4 (user, before callback)
                                        ;     [ESP](user) = EIP (user) before callback
                                        ;
                                        ; Note: If CPU was in user mode while entering
                                        ;     the timer interrupt service routine,
                                        ;     'IRET' will get return to callback routine
```

```
1907                                    ;      immediately. If CPU was in system/kernel mode
1908                                    ;      'iret' will get return to system call and
1909                                    ;      then, callback routine will be return address
1910                                    ;      from system call. (User's callback/service code
1911                                    ;      will be able to return to normal return address
1912                                    ;      via an 'retn' at the end.)
1913                                    ;
1914                                    ; Note(**): User's callback service code must be ended
1915                                    ;      with a 'sysrele' sytstem call ! (09/01/2017)
1916                                    ;
1917                                    ;      For example:
1918                                    ;
1919                                    ;      timer_callback:
1920                                    ;          ...
1921                                    ;          inc      dword [time_counter]
1922                                    ;          ...
1923                                    ;          mov eax, 39 ; 'sysrele'
1924                                    ;          int 40h ; TRDOS 386 system call (interrupt)
1925                                    ;
1926                                    ;
1927                                    ;; Note(*): User's callback service code must preserve cpu
1928                                    ;;     flags if it has any instructions which changes
1929                                    ;;     flags in the service code. (25/12/2016)
1930                                    ;;
1931                                    ;;     For example:
1932                                    ;;
1933                                    ;;     timer_callback:
1934                                    ;;         pushf ; save flags
1935                                    ;;         ; this instruction changes zero flag
1936                                    ;;         inc      dword [time_counter]
1937                                    ;;         popf ; restore flags
1938                                    ;;         retn ; return to normal user code
1939                                    ;;              (which is interrupted by the
1940                                    ;;               timer interput)
1941                                    ;;
1942
1943                                    ; 15/01/2017
1944 00000AAD 8B2D[5C030300]             mov    ebp, [u.sp]; kernel's stack, points to EIP (user)
1945 00000AB3 895D00                   mov    [ebp], ebx
1946 00000AB6 E95CFFFFFF               jmp    timer_int_return
1947
1948                                    ; 15/01/2017
1949                                    ; 13/01/2017
1950                                    ; 19/12/2016
1951                                    ; 06/06/2016
1952                                    ; 23/05/2016
1953                                    ; 22/05/2016
1954                                    ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
1955                                    ; 26/02/2015
1956                                    ; 07/09/2014
1957                                    ; 25/08/2014
1958                                rtc_int:       ; Real Time Clock Interrupt (IRQ 8)
1959                                    ; 22/05/2016
1960 00000ABB 1E                       push   ds ; ** ; 23/05/2016
1961 00000ABC 50                       push   eax ; *
1962 00000ABD 66B81000                 mov    ax, KDATA
1963 00000AC1 8ED8                     mov    ds, ax
1964                                    ;
1965 00000AC3 8A25[26810100]           mov    ah, [RTC_2Hz] ;  2 Hz interrupt to 1 Hz function
1966 00000AC9 80F401                   xor    ah, 1
1967 00000ACC 8825[26810100]           mov    [RTC_2Hz], ah ; 1 = 0.5 second, 0 = 1 second
1968 00000AD2 753B                     jnz    short rtc_int_return ; half second
1969                                    ; 1 second
1970                                rtc_int_0:
1971                                    ; 22/05/2016
1972 00000AD4 58                       pop    eax ; *
1973                                    ;
1974                                    ; 14/10/2015 ('timer_int')
1975                                    ; Here, we are simulating system call entry (for task switch)
1976                                    ; (If multitasking is enabled,
1977                                    ; 'clock' procedure may jump to 'sysrelease')
1978                                    ;push  ds ; ** ; 23/05/2016
1979 00000AD5 06                       push   es
1980 00000AD6 0FA0                     push   fs
1981 00000AD8 0FA8                     push   gs
1982 00000ADA 60                       pushad  ; eax, ecx, edx, ebx, esp -before pushad-, ebp, esi, edi
1983 00000ADB 66B91000                 mov    cx, KDATA
1984                                    ;mov  ds, cx ; 06/06/2016
1985 00000ADF 8EC1                     mov    es, cx
1986 00000AE1 8EE1                     mov    fs, cx
1987 00000AE3 8EE9                     mov    gs, cx
1988                                    ;
1989 00000AE5 0F20D9                   mov    ecx, cr3
1990 00000AE8 890D[5C040300]           mov    [cr3reg], ecx ; save current cr3 register value/content
1991                                    ;
1992 00000AEE 803D[D4030300]00         cmp    byte [u.t_lock], 0 ; timer lock (callback) status ?
1993 00000AF5 7711                     ja     short rtc_int_1        ; yes
1994
1995                                    ; 15/01/2017
1996 00000AF7 3B0D[A8800100]           cmp    ecx, [k_page_dir]
1997 00000AFD 7409                     je     short rtc_int_1
1998
1999 00000AFF 8B0D[A8800100]           mov    ecx, [k_page_dir]
2000 00000B05 0F22D9                   mov    cr3, ecx
2001                                rtc_int_1:
2002                                    ; Timer event (kernel) functions must be performed with
2003                                    ; 1 second intervals - TRDOS 386 (TRDOS v2.0) feature ! -
2004                                    ;
2005                                    ; 25/08/2014
2006 00000B08 E818030000               call   rtc_p  ; 19/05/2016 - major modification
2007
2008                                    ; 23/05/2016
2009 00000B0D 28E4                     sub    ah, ah ; 0
2010                                    ; 22/05/2016 - TRDOS 386 timer event modifications
2011                                rtc_int_return: ; 19/05/2016
```

```
                                        ; 22/02/2015 - dsectpm.s
                                        ; [ source: http://wiki.osdev.org/RTC ]
                                        ; read status register C to complete procedure
                                        ;(it is needed to get a next IRQ 8)
00000B0F B00C                           mov    al, 0Ch ;
00000B11 E670                           out    70h, al ; select register C
00000B13 90                             nop
00000B14 E471                           in     al, 71h ; just throw away contents
                                        ; 22/02/2015
00000B16 B020                           MOV    AL,EOI       ; END OF INTERRUPT
                                        ;CLI                ; DISABLE INTERRUPTS TILL STACK CLEARED
00000B18 E6A0                           OUT    INTB00,AL    ; FOR CONTROLLER #2

                                        ; 23/05/2016
00000B1A B020                           MOV    AL,EOI       ; GET END OF INTERRUPT MASK
00000B1C FA                             CLI                 ; DISABLE INTERRUPTS TILL STACK CLEARED
00000B1D E620                           OUT    INTA00,AL    ; END OF INTERRUPT TO 8259 - 1
                                        ;
                                        ; 23/05/2016
00000B1F 20E4                           and    ah, ah
00000B21 0F84B7FEFFFF                   jz     rtc_int_2

                                        ; ah = 1 (half second)
00000B27 58                             pop    eax ; *
00000B28 1F                             pop    ds  ; **
00000B29 CF                             iretd

                                ; ////////////////

                                        ; 28/08/2014
                                irq0:
00000B2A 6A00                           push   dword 0
00000B2C EB48                           jmp    short which_irq
                                irq1:
00000B2E 6A01                           push   dword 1
00000B30 EB44                           jmp    short which_irq
                                irq2:
00000B32 6A02                           push   dword 2
00000B34 EB40                           jmp    short which_irq
                                irq3:
                                        ; 20/11/2015
                                        ; 24/10/2015
00000B36 2EFF15[B8290100]               call   dword [cs:com2_irq3]
00000B3D 6A03                           push   dword 3
00000B3F EB35                           jmp    short which_irq
                                irq4:
                                        ; 20/11/2015
                                        ; 24/10/2015
00000B41 2EFF15[B4290100]               call   dword [cs:com1_irq4]
00000B48 6A04                           push   dword 4
00000B4A EB2A                           jmp    short which_irq
                                irq5:
00000B4C 6A05                           push   dword 5
00000B4E EB26                           jmp    short which_irq
                                irq6:
00000B50 6A06                           push   dword 6
00000B52 EB22                           jmp    short which_irq
                                irq7:
00000B54 6A07                           push   dword 7
00000B56 EB1E                           jmp    short which_irq
                                irq8:
00000B58 6A08                           push   dword 8
00000B5A EB1A                           jmp    short which_irq
                                irq9:
00000B5C 6A09                           push   dword 9
00000B5E EB16                           jmp    short which_irq
                                irq10:
00000B60 6A0A                           push   dword 10
00000B62 EB12                           jmp    short which_irq
                                irq11:
00000B64 6A0B                           push   dword 11
00000B66 EB0E                           jmp    short which_irq
                                irq12:
00000B68 6A0C                           push   dword 12
00000B6A EB0A                           jmp    short which_irq
                                irq13:
00000B6C 6A0D                           push   dword 13
00000B6E EB06                           jmp    short which_irq
                                irq14:
00000B70 6A0E                           push   dword 14
00000B72 EB02                           jmp    short which_irq
                                irq15:
00000B74 6A0F                           push   dword 15
                                        ;jmp   short which_irq

                                        ; 22/01/2017
                                        ; 19/10/2015
                                        ; 29/08/2014
                                        ; 21/08/2014
                                which_irq:
00000B76 870424                         xchg   eax, [esp]  ; 28/08/2014
00000B79 53                             push   ebx
00000B7A 56                             push   esi
00000B7B 57                             push   edi
00000B7C 1E                             push   ds
00000B7D 06                             push   es
                                        ;
00000B7E 88C3                           mov    bl, al
                                        ;
00000B80 B810000000                     mov    eax, KDATA
00000B85 8ED8                           mov    ds, ax
00000B87 8EC0                           mov    es, ax
                                        ; 19/10/2015
00000B89 FC                             cld
                                          ; 27/08/2014
```

```
2117 00000B8A 8105[B0400100]A000-          add    dword [scr_row], 0A0h
2117 00000B92 0000
2118                                       ;
2119 00000B94 B417                         mov    ah, 17h      ; blue (1) background,
2120                                                   ; light gray (7) forecolor
2121 00000B96 8B3D[B0400100]               mov    edi, [scr_row]
2122 00000B9C B049                         mov    al, 'I'
2123 00000B9E 66AB                         stosw
2124 00000BA0 B052                         mov    al, 'R'
2125 00000BA2 66AB                         stosw
2126 00000BA4 B051                         mov    al, 'Q'
2127 00000BA6 66AB                         stosw
2128 00000BA8 B020                         mov    al, ' '
2129 00000BAA 66AB                         stosw
2130 00000BAC 88D8                         mov    al, bl
2131 00000BAE 3C0A                         cmp    al, 10
2132 00000BB0 7208                         jb     short ii1
2133 00000BB2 B031                         mov    al, '1'
2134 00000BB4 66AB                         stosw
2135 00000BB6 88D8                         mov    al, bl
2136 00000BB8 2C0A                         sub    al, 10
2137                                  ii1:
2138 00000BBA 0430                         add    al, '0'
2139 00000BBC 66AB                         stosw
2140 00000BBE B020                         mov    al, ' '
2141 00000BC0 66AB                         stosw
2142 00000BC2 B021                         mov    al, '!'
2143 00000BC4 66AB                         stosw
2144 00000BC6 B020                         mov    al, ' '
2145 00000BC8 66AB                         stosw
2146                                       ; 23/02/2015
2147 00000BCA 80FB07                       cmp    bl, 7 ; check for IRQ 8 to IRQ 15
2148 00000BCD 7604                         jna    ii2
2149                                       ; 22/01/2017
2150 00000BCF B020                         mov    al, 20h  ; END OF INTERRUPT COMMAND TO
2151 00000BD1 E6A0                         out    0A0h, al ; the 2nd 8259
2152                                  ii2:
2153 00000BD3 B020                         mov    al, 20h  ; END OF INTERRUPT COMMAND TO
2154 00000BD5 E620                         out    20h, al ; the 2nd 8259
2155 00000BD7 E9CB010000                   jmp    iiret
2156                                       ;
2157                                       ; 22/08/2014
2158                                       ;mov   al, 20h ; END OF INTERRUPT COMMAND TO 8259
2159                                       ;out   20h, al      ; 8259 PORT
2160                                       ;
2161                                       ;pop   es
2162                                       ;pop   ds
2163                                       ;pop   edi
2164                                       ;pop   esi
2165                                       ;pop   ebx
2166                                       ;pop   eax
2167                                       ;iret
2168
2169                                       ; 02/04/2015
2170                                       ; 25/08/2014
2171                                  exc0:
2172 00000BDC 6A00                         push   dword 0
2173 00000BDE E990000000                   jmp    cpu_except
2174                                  exc1:
2175 00000BE3 6A01                         push   word 1
2176 00000BE5 E989000000                   jmp    cpu_except
2177                                  exc2:
2178 00000BEA 6A02                         push   dword 2
2179 00000BEC E982000000                   jmp    cpu_except
2180                                  exc3:
2181 00000BF1 6A03                         push   dword 3
2182 00000BF3 EB7E                         jmp    cpu_except
2183                                  exc4:
2184 00000BF5 6A04                         push   dword 4
2185 00000BF7 EB7A                         jmp    cpu_except
2186                                  exc5:
2187 00000BF9 6A05                         push   dword 5
2188 00000BFB EB76                         jmp    cpu_except
2189                                  exc6:
2190 00000BFD 6A06                         push   dword 6
2191 00000BFF EB72                         jmp    cpu_except
2192                                  exc7:
2193 00000C01 6A07                         push   dword 7
2194 00000C03 EB6E                         jmp    cpu_except
2195                                  exc8:
2196                                       ; [esp] = Error code
2197 00000C05 6A08                         push   dword 8
2198 00000C07 EB5C                         jmp    cpu_except_en
2199                                  exc9:
2200 00000C09 6A09                         push   dword 9
2201 00000C0B EB66                         jmp    cpu_except
2202                                  exc10:
2203                                       ; [esp] = Error code
2204 00000C0D 6A0A                         push   dword 10
2205 00000C0F EB54                         jmp    cpu_except_en
2206                                  exc11:
2207                                       ; [esp] = Error code
2208 00000C11 6A0B                         push   dword 11
2209 00000C13 EB50                         jmp    cpu_except_en
2210                                  exc12:
2211                                       ; [esp] = Error code
2212 00000C15 6A0C                         push   dword 12
2213 00000C17 EB4C                         jmp    cpu_except_en
2214                                  exc13:
2215                                       ; [esp] = Error code
2216 00000C19 6A0D                         push   dword 13
2217 00000C1B EB48                         jmp    cpu_except_en
2218                                  exc14:
2219                                       ; [esp] = Error code
2220 00000C1D 6A0E                         push   dword 14
```

```
2221 00000C1F EB44                      jmp    short cpu_except_en
2222                                 exc15:
2223 00000C21 6A0F                      push   dword 15
2224 00000C23 EB4E                      jmp    cpu_except
2225                                 exc16:
2226 00000C25 6A10                      push   dword 16
2227 00000C27 EB4A                      jmp    cpu_except
2228                                 exc17:
2229                                    ; [esp] = Error code
2230 00000C29 6A11                      push   dword 17
2231 00000C2B EB38                      jmp    short cpu_except_en
2232                                 exc18:
2233 00000C2D 6A12                      push   dword 18
2234 00000C2F EB42                      jmp    short cpu_except
2235                                 exc19:
2236 00000C31 6A13                      push   dword 19
2237 00000C33 EB3E                      jmp    short cpu_except
2238                                 exc20:
2239 00000C35 6A14                      push   dword 20
2240 00000C37 EB3A                      jmp    short cpu_except
2241                                 exc21:
2242 00000C39 6A15                      push   dword 21
2243 00000C3B EB36                      jmp    short cpu_except
2244                                 exc22:
2245 00000C3D 6A16                      push   dword 22
2246 00000C3F EB32                      jmp    short cpu_except
2247                                 exc23:
2248 00000C41 6A17                      push   dword 23
2249 00000C43 EB2E                      jmp    short cpu_except
2250                                 exc24:
2251 00000C45 6A18                      push   dword 24
2252 00000C47 EB2A                      jmp    short cpu_except
2253                                 exc25:
2254 00000C49 6A19                      push   dword 25
2255 00000C4B EB26                      jmp    short cpu_except
2256                                 exc26:
2257 00000C4D 6A1A                      push   dword 26
2258 00000C4F EB22                      jmp    short cpu_except
2259                                 exc27:
2260 00000C51 6A1B                      push   dword 27
2261 00000C53 EB1E                      jmp    short cpu_except
2262                                 exc28:
2263 00000C55 6A1C                      push   dword 28
2264 00000C57 EB1A                      jmp    short cpu_except
2265                                 exc29:
2266 00000C59 6A1D                      push   dword 29
2267 00000C5B EB16                      jmp    short cpu_except
2268                                 exc30:
2269 00000C5D 6A1E                      push   dword 30
2270 00000C5F EB04                      jmp    short cpu_except_en
2271                                 exc31:
2272 00000C61 6A1F                      push   dword 31
2273 00000C63 EB0E                      jmp    short cpu_except
2274
2275                                    ; 19/10/2015
2276                                    ; 19/09/2015
2277                                    ; 01/09/2015
2278                                    ; 28/08/2015
2279                                    ; 28/08/2014
2280                                 cpu_except_en:
2281 00000C65 87442404                  xchg   eax, [esp+4] ; Error code
2282 00000C69 36A3[60050300]            mov    [ss:error_code], eax
2283 00000C6F 58                        pop    eax  ; Exception number
2284 00000C70 870424                    xchg   eax, [esp]
2285                                           ; eax = eax before exception
2286                                           ; [esp] -> exception number
2287                                           ; [esp+4] -> EIP to return
2288                                    ; 22/01/2017
2289                                    ; 19/10/2015
2290                                    ; 19/09/2015
2291                                    ; 01/09/2015
2292                                    ; 28/08/2015
2293                                    ; 29/08/2014
2294                                    ; 28/08/2014
2295                                    ; 25/08/2014
2296                                    ; 21/08/2014
2297                                 cpu_except:  ; CPU Exceptions
2298 00000C73 FC                        cld
2299 00000C74 870424                    xchg   eax, [esp]
2300                                           ; eax = Exception number
2301                                           ; [esp] = eax (before exception)
2302 00000C77 53                        push   ebx
2303 00000C78 56                        push   esi
2304 00000C79 57                        push   edi
2305 00000C7A 1E                        push   ds
2306 00000C7B 06                        push   es
2307                                    ; 28/08/2015
2308 00000C7C 66BB1000                  mov    bx, KDATA
2309 00000C80 8EDB                      mov    ds, bx
2310 00000C82 8EC3                      mov    es, bx
2311 00000C84 0F20DB                    mov    ebx, cr3
2312 00000C87 53                        push   ebx ; (*) page directory
2313                                    ; 19/10/2015
2314 00000C88 FC                        cld
2315                                    ; 25/03/2015
2316 00000C89 8B1D[A8800100]            mov    ebx, [k_page_dir]
2317 00000C8F 0F22DB                    mov    cr3, ebx
2318                                    ; 28/08/2015
2319 00000C92 83F80E                    cmp    eax, 0Eh ; 14, PAGE FAULT
2320 00000C95 750F                      jne    short cpu_except_nfp
2321 00000C97 E889500000                call   page_fault_handler
2322 00000C9C 21C0                      and    eax, eax
2323 00000C9E 0F84FF000000              jz     iiretp ; 01/09/2015
2324 00000CA4 B00E                      mov    al, 0Eh ; 14
2325                                 cpu_except_nfp:
```

```
2326                                    ; 23/08/2016
2327 00000CA6 803D[BA6A0000]03          cmp   byte [CRT_MODE], 3
2328 00000CAD 7409                      je    short cpu_except_mode_3
2329 00000CAF 50                        push  eax
2330 00000CB0 B003                      mov   al, 3
2331 00000CB2 E8430E0000                call  _set_mode
2332 00000CB7 58                        pop   eax
2333                                cpu_except_mode_3:
2334                                    ; 02/04/2015
2335 00000CB8 BB[09090000]              mov   ebx, hang
2336 00000CBD 875C241C                  xchg  ebx, [esp+28]
2337                                          ; EIP (points to instruction which faults)
2338                                          ; New EIP (hang)
2339 00000CC1 891D[64050300]            mov   [FaultOffset], ebx
2340 00000CC7 C744242008000000          mov   dword [esp+32], KCODE ; kernel's code segment
2341 00000CCF 814C242400020000          or    dword [esp+36], 200h ; enable interrupts (set IF)
2342                                    ;
2343 00000CD7 88C4                      mov   ah, al
2344 00000CD9 240F                      and   al, 0Fh
2345 00000CDB 3C09                      cmp   al, 9
2346 00000CDD 7602                      jna   short h1ok
2347 00000CDF 0407                      add   al, 'A'-':'
2348                                h1ok:
2349 00000CE1 C0EC04                    shr   ah, 4
2350 00000CE4 80FC09                    cmp   ah, 9
2351 00000CE7 7603                      jna   short h2ok
2352 00000CE9 80C407                    add   ah, 'A'-':'
2353                                h2ok:
2354 00000CEC 86E0                      xchg  ah, al
2355 00000CEE 66053030                  add   ax, '00'
2356 00000CF2 66A3[08430100]            mov   [excnstr], ax
2357                                    ;
2358                                    ; 29/08/2014
2359 00000CF8 A1[64050300]              mov   eax, [FaultOffset]
2360 00000CFD 51                        push  ecx
2361 00000CFE 52                        push  edx
2362 00000CFF 89E3                      mov   ebx, esp
2363                                    ; 28/08/2015
2364 00000D01 B910000000                mov   ecx, 16  ; divisor value to convert binary number
2365                                                   ; to hexadecimal string
2366                                    ;mov  ecx, 10    ; divisor to convert
2367                                                     ; binary number to decimal string
2368                                b2d1:
2369 00000D06 31D2                      xor   edx, edx
2370 00000D08 F7F1                      div   ecx
2371                                    ;push dx
2372                                    ; 18/04/2021
2373 00000D0A 52                        push  edx
2374 00000D0B 39C8                      cmp   eax, ecx
2375 00000D0D 73F7                      jnb   short b2d1
2376 00000D0F BF[13430100]              mov   edi, EIPstr ; EIP value
2377                                                      ; points to instruction which faults
2378                                    ; 28/08/2015
2379 00000D14 89C2                      mov   edx, eax
2380                                b2d2:
2381                                    ;add  al, '0'
2382 00000D16 8A82[5D420000]            mov   al, [edx+hexchrs]
2383 00000D1C AA                        stosb           ; write hexadecimal digit to its place
2384 00000D1D 39E3                      cmp   ebx, esp
2385 00000D1F 7605                      jna   short b2d3
2386                                    ;pop  ax
2387                                    ; 18/04/2021
2388 00000D21 58                        pop   eax
2389 00000D22 88C2                      mov   dl, al
2390 00000D24 EBF0                      jmp   short b2d2
2391                                b2d3:
2392 00000D26 B068                      mov   al, 'h' ; 28/08/2015
2393 00000D28 AA                        stosb
2394 00000D29 B020                      mov   al, 20h          ; space
2395 00000D2B AA                        stosb
2396 00000D2C 30C0                      xor   al, al     ; to do it an ASCIIZ string
2397 00000D2E AA                        stosb
2398                                    ;
2399 00000D2F 5A                        pop   edx
2400 00000D30 59                        pop   ecx
2401                                    ;
2402 00000D31 B44F                      mov   ah, 4Fh      ; red (4) background,
2403                                                  ; white (F) forecolor
2404 00000D33 BE[F8420100]              mov   esi, exc_msg ; message offset
2405                                    ;
2406                                    ; 20/01/2017 (!cpu exception!)
2407                                    ;
2408 00000D38 8105[B0400100]A000-       add   dword [scr_row], 0A0h
2408 00000D40 0000
2409 00000D42 8B3D[B0400100]            mov   edi, [scr_row]
2410                                    ;
2411 00000D48 C605[5B030300]00          mov   byte [sysflg], 0  ; system mode
2412 00000D4F FB                        sti
2413                                    ;
2414 00000D50 E8EFFBFFFF                call  printk
2415                                    ;
2416 00000D55 B410                      mov   ah, 10h
2417 00000D57 E87F010000                call  int16h ; getc
2418                                    ;
2419 00000D5C B003                      mov   al, 3
2420 00000D5E E8970D0000                call  _set_mode
2421                                    ;
2422 00000D63 B801000000                mov   eax, 1
2423 00000D68 E9EAC70000                jmp   sysexit ; terminate process !!!
2424
2425                                    ; 22/01/2017
2426                                    ; 18/04/2016
2427                                    ; 28/08/2015
2428                                    ; 23/02/2015
2429                                    ; 20/08/2014
```

```
                                ignore_int:
00000D6D 50                             push   eax
00000D6E 53                             push   ebx ; 23/02/2015
00000D6F 56                             push   esi
00000D70 57                             push   edi
00000D71 1E                             push   ds
00000D72 06                             push   es
                                        ; 18/04/2016
00000D73 66B81000                       mov    ax, KDATA
00000D77 8ED8                           mov    ds, ax
00000D79 8EC0                           mov    es, ax
                                        ; 28/08/2015
00000D7B 0F20D8                         mov    eax, cr3
00000D7E 50                             push   eax ; (*) page directory
                                        ;
00000D7F B467                           mov    ah, 67h      ; brown (6) background,
                                                     ; light gray (7) forecolor
00000D81 BE[C0410100]                   mov    esi, int_msg ; message offset
                                piemsg:
                                          ; 27/08/2014
00000D86 8105[B0400100]A000-            add   dword [scr_row], 0A0h
00000D8E 0000
00000D90 8B3D[B0400100]                 mov   edi, [scr_row]
                                          ;
00000D96 E8A9FBFFFF                     call   printk
                                        ;
                                        ; 23/02/2015
00000D9B B020                           mov    al, 20h  ; END OF INTERRUPT COMMAND TO
00000D9D E6A0                           out    0A0h, al ; the 2nd 8259
                                        ; 22/08/2014
00000D9F B020                           mov    al, 20h ; END OF INTERRUPT COMMAND TO 8259
00000DA1 E620                           out    20h, al      ; 8259 PORT
                                iiretp:
                                        ; 22/01/2017
                                        ; 01/09/2015
                                        ; 28/08/2015
00000DA3 58                             pop    eax ; (*) page directory
00000DA4 0F22D8                         mov    cr3, eax
                                iiret:
00000DA7 07                             pop    es
00000DA8 1F                             pop    ds
00000DA9 5F                             pop    edi
00000DAA 5E                             pop    esi
00000DAB 5B                             pop    ebx ; 29/08/2014
00000DAC 58                             pop    eax
00000DAD CF                             iretd
                                
                                        ; 23/05/2016
                                        ; 22/08/2014
                                        ; IBM PC/AT BIOS source code ----- 10/06/85 (bios.asm)
                                        ; (INT 1Ah)
                                        ;; Linux (v0.12) source code (main.c) by Linus Torvalds (1991)
                                time_of_day:
00000DAE E8C1590000                     call   UPD_IPR                    ; WAIT TILL UPDATE NOT IN PROGRESS
00000DB3 726F                           jc     short time_of_day_retn ; 23/05/2016
00000DB5 B000                           mov    al, CMOS_SECONDS
00000DB7 E8EE590000                     call   CMOS_READ
00000DBC A2[18810100]                   mov    [time_seconds], al
00000DC1 B002                           mov    al, CMOS_MINUTES
00000DC3 E8E2590000                     call   CMOS_READ
00000DC8 A2[19810100]                   mov    [time_minutes], al
00000DCD B004                           mov    al, CMOS_HOURS
00000DCF E8D6590000                     call   CMOS_READ
00000DD4 A2[1A810100]                   mov   [time_hours], al
00000DD9 B006                           mov    al, CMOS_DAY_WEEK
00000DDB E8CA590000                     call   CMOS_READ
00000DE0 A2[1B810100]                   mov    [date_wday], al
00000DE5 B007                           mov    al, CMOS_DAY_MONTH
00000DE7 E8BE590000                     call   CMOS_READ
00000DEC A2[1C810100]                   mov    [date_day], al
00000DF1 B008                           mov    al, CMOS_MONTH
00000DF3 E8B2590000                     call   CMOS_READ
00000DF8 A2[1D810100]                   mov    [date_month], al
00000DFD B009                           mov    al, CMOS_YEAR
00000DFF E8A6590000                     call   CMOS_READ
00000E04 A2[1E810100]                   mov    [date_year], al
00000E09 B032                           mov    al, CMOS_CENTURY
00000E0B E89A590000                     call   CMOS_READ
00000E10 A2[1F810100]                   mov    [date_century], al
                                        ;
00000E15 B000                           mov    al, CMOS_SECONDS
00000E17 E88E590000                     call   CMOS_READ
00000E1C 3A05[18810100]                 cmp    al, [time_seconds]
00000E22 758A                           jne    short time_of_day
                                
                                time_of_day_retn:
00000E24 C3                             retn
                                
                                        ; 15/01/2017
                                        ; 10/06/2016
                                        ; 07/06/2016
                                        ; 06/06/2016
                                        ; 23/05/2016
                                rtc_p:
00000E25 B101                           mov    cl, 1 ; 15/01/2017
00000E27 EB02                           jmp    short rtc_p0
                                u_timer:
                                        ; Timer Events with 18.2 Hz Timer Ticks
                                        ; (and also timer events with RTC seconds)
00000E29 28C9                           sub    cl, cl ; mov cl, 0 ; 15/01/2017
                                rtc_p0:
                                        ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
                                        ; Major Modification:
                                        ; Check and Perform Timer Events (for RTC)
                                        ; 25/08/2014 - 07/09/2014
```

```
                                        ; Retro UNIX 386 v1:
                                        ; Print Real Time Clock content
                                        ;
                                        ; 15/01/2017
00000E2B 880D[388D0100]                 mov   byte [priority], cl ; 0 or 1 (not 2)
00000E31 8A2D[3B8D0100]                 mov   ch, [timer_events]
00000E37 20ED                           and   ch, ch
00000E39 7420                           jz    short rtc_p3
                                        ;
00000E3B BE[60040300]                   mov   esi, timer_set  ; beginning address of
                                                              ; timer events space
                                rtc_p1:
00000E40 8B06                           mov   eax, [esi]
00000E42 20C0                           and   al, al ; 0 = free, >0 = process no.
00000E44 7416                           jz    short rtc_p4
                                        ;
00000E46 C1C810                         ror   eax, 16
                                        ; ah = response value, al = interrupt type
                                        ; 15/01/2017
                                        ; cl = interrupt source
                                        ;       1 = RTC, 0 = PIT
00000E49 38C8                           cmp   al, cl
00000E4B 750A                           jne   short rtc_p2 ; not as requested or undefined !
00000E4D 3C01                           cmp   al, 1 ; 1 ; RTC interrupt ?
00000E4F 7410                           je    short rtc_p5 ; yes, check for response
                                        ; 06/06/2016 - 18.2 Hz Timer Ticks
00000E51 836E080A                       sub   dword [esi+8], 10 ; 1 tick = 10
00000E55 7613                           jna   short rtc_p6  ; continue for responding
                                rtc_p2:
                                        ; 15/01/2017 (cl -> ch)
                                        ; 07/06/2016
00000E57 FECD                           dec   ch    ; remain count of timer events
00000E59 7501                           jnz   short rtc_p4
                                rtc_p3:
00000E5B C3                             retn
                                rtc_p4:
                                        ;cmp   esi, timer_set + 240 ; 15*16 (last event)
                                        ;jnb   short rtc_p3 ; end of timer event space
00000E5C 83C610                         add   esi, 16 ; next timer event
00000E5F EBDF                           jmp   short rtc_p1
                                rtc_p5:
                                        ; current timer count ; 06/06/2016 (182)
00000E61 816E08B6000000                 sub   dword [esi+8], 182 ; 1 second (10*18.2)
00000E68 77ED                           ja    short rtc_p2  ; check for the next
                                rtc_p6:
                                        ; it is the time of response!
00000E6A 8B5E04                         mov   ebx, [esi+4] ; set (count limit) value
00000E6D 895E08                         mov   [esi+8], ebx ; reset count down value
                                                           ; to count limit
                                        ; 19/12/2016
                                        ; 10/12/2016 - timer callback modification
00000E70 8B7E0C                         mov   edi, [esi+12] ; response (or callback) address
00000E73 807E0100                       cmp   byte [esi+1], 0 ; >0 = callback
00000E77 762A                           jna   short rtc_p8
                                        ;
                                        ; timer callback !
00000E79 0FB61E                         movzx ebx, byte [esi] ; process number (>0)
00000E7C 89D8                           mov   eax, ebx
00000E7E C0E302                         shl   bl, 2 ; *4
00000E81 89BB[0C010300]                 mov   [ebx+p.tcb-4], edi ; user's callback service addr
00000E87 3A05[B3030300]                 cmp   al, [u.uno]
00000E8D 7521                           jne   short rtc_p9
00000E8F 893D[D0030300]                 mov   [u.tcb], edi
                                rtc_p7:
                                        ; 15/01/2017
00000E95 B002                           mov   al, 2
00000E97 A2[388D0100]                   mov   [priority], al ; 2
                                        ; 10/01/2017
                                        ;mov   byte [u.pri], 2
00000E9C A2[A9030300]                   mov   [u.pri], al ; 2
00000EA1 EBB4                           jmp   short rtc_p2
                                rtc_p8:
                                        ; response address is physical address of
                                        ; the program's response (signal return) byte
                                        ; 06/06/2016
                                        ;mov   edi, [esi+12] ; response address
00000EA3 8827                           mov   [edi], ah     ; response value
                                        ;
00000EA5 C1C010                         rol   eax, 16
                                        ; 15/01/2017
00000EA8 3A05[B3030300]                 cmp   al, [u.uno] ; running process ?
00000EAE 74E5                           je    short rtc_p7
                                rtc_p9:
                                        ; al = process number  ; 10/06/2016
00000EB0 B202                           mov   dl, 2 ; priority, 2 = event (high)
00000EB2 E8F7170100                     call  set_run_sequence ; 19/05/2016
00000EB7 EB9E                           jmp   short rtc_p2 ; 10/06/2016
                                
                                ; Default IRQ 7 handler against spurious IRQs (from master PIC)
                                ; 25/02/2015 (source: http://wiki.osdev.org/8259_PIC)
                                default_irq7:
                                        ;push  ax
                                        ; 18/04/2021
00000EB9 50                             push  eax
00000EBA B00B                           mov   al, 0Bh  ; In-Service register
00000EBC E620                           out   20h, al
00000EBE EB00                             jmp short $+2
00000EC0 EB00                           jmp short $+2
00000EC2 E420                           in    al, 20h
00000EC4 2480                           and   al, 80h ; bit 7 (is it real IRQ 7 or fake?)
00000EC6 7404                             jz      short irq7_iret ; Fake (spurious) IRQ, do not send EOI
00000EC8 B020                             mov     al, 20h ; EOI
00000ECA E620                           out   20h, al
                                irq7_iret:
                                        ;pop   ax
```

```
                                        ; 18/04/2021
00000ECC 58                             pop    eax
00000ECD CF                             iretd

                                bcd_to_ascii:
                                        ; 25/08/2014
                                        ; INPUT ->
                                        ;      al = Packed BCD number
                                        ; OUTPUT ->
                                        ;      ax  = ASCII word/number
                                        ;
                                        ; Erdogan Tan - 1998 (proc_hex) - TRDOS.ASM (2004-2011)
                                        ;
00000ECE D410                           db     0D4h, 10h     ; Undocumented inst. AAM
                                                             ; AH = AL / 10h
                                                             ; AL = AL MOD 10h
00000ED0 660D3030                       or     ax, '00'      ; Make it ASCII based

00000ED4 86E0                           xchg   ah, al

00000ED6 C3                             retn

                                ; 15/12/2020
                                real_mem_16m_64k:
00000ED7 0000                           dw     0      ; Real size of system memory (if > 16MB)
                                                      ; as number of 64K blocks - 256
                                                      ; (This is for saving real system memory
                                                      ; because if system memory is larger than
                                                      ; 3 GB and if a VESA VBE video bios
                                                      ; is detected, 'mem_16m_64K' may be
                                                      ; decreased to reserve LFB space
                                                      ; at the end of system memory.)
                                                      ; Upper memory space from LFB base address
                                                      ; to 4GB will not be included by M.A.T.
                                    def_LFB_addr:
00000ED9 0000                             dw     0      ; HW of default LFB addr (for mode 118h)


                                    %include 'keyboard.s' ; 07/03/2015
                                <1> ; ****************************************************************************
                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.3 - keyboard.s
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Last Update: 12/04/2021
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Beginning: 17/01/2016
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Assembler: NASM version 2.15 (trdos386.s)
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Turkish Rational DOS
                                <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
                                <1> ;
                                <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
                                <1> ; keyboard.inc (17/10/2015)
                                <1> ;
                                <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
                                <1> ; ****************************************************************************
                                <1>
                                <1> ; Retro UNIX 386 v1 Kernel - KEYBOARD.INC
                                <1> ; Last Modification: 17/10/2015
                                <1> ;                (Keyboard Data is in 'KYBDATA.INC')
                                <1> ;
                                <1> ; ///////// KEYBOARD FUNCTIONS (PROCEDURES) ///////////////
                                <1>
                                <1> ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>
                                <1> ; 03/12/2014
                                <1> ; 26/08/2014
                                <1> ; KEYBOARD I/O
                                <1> ; (INT_16h - Retro UNIX 8086 v1 - U9.ASM, 30/06/2014)
                                <1>
                                <1> ;NOTE: 'k0' to 'k7' are name of OPMASK registers.
                                <1> ;     (The reason of using '_k' labels!!!) (27/08/2014)
                                <1> ;NOTE: 'NOT' keyword is '~' unary operator in NASM.
                                <1> ;     ('NOT LC_HC' --> '~LC_HC') (bit reversing operator)
                                <1>
                                <1> int16h:     ; 30/06/2015
                                <1> ;getc:
00000EDB 9C                     <1>     pushfd ; 28/08/2014
00000EDC 0E                     <1>     push   cs
00000EDD E801000000             <1>     call   KEYBOARD_IO_1 ; getc_int
00000EE2 C3                     <1>     retn
                                <1>
                                <1> getc_int:
                                <1>     ; 28/02/2015
                                <1>     ; 03/12/2014 (derivation from pc-xt-286 bios source code -1986-,
                                <1>     ;           instead of pc-at bios - 1985-)
                                <1>     ; 28/08/2014 (_k1d)
                                <1>     ; 30/06/2014
                                <1>     ; 03/03/2014
                                <1>     ; 28/02/2014
                                <1>     ; Derived from "KEYBOARD_IO_1" procedure of IBM "pc-xt-286"
                                <1>     ; rombios source code (21/04/1986)
                                <1>     ;      'keybd.asm', INT 16H, KEYBOARD_IO
                                <1>     ;
                                <1>     ; KYBD --- 03/06/86  KEYBOARD BIOS
                                <1>     ;
                                <1>     ;--- INT 16 H ----------------------------------------------------------------
                                <1>     ; KEYBOARD I/O                                                          :
                                <1>     ;     THESE ROUTINES PROVIDE READ KEYBOARD SUPPORT                       :
                                <1>     ; INPUT                                                                 :
                                <1>     ;     (AH)= 00H  READ THE NEXT ASCII CHARACTER ENTERED FROM THE KEYBOARD,  :
                                <1>     ;              RETURN THE RESULT IN (AL), SCAN CODE IN (AH).            :
                                <1>     ;              THIS IS THE COMPATIBLE READ INTERFACE, EQUIVALENT TO THE  :
                                <1>     ;                 STANDARD PC OR PCAT KEYBOARD                          :
                                <1>     ;-------------------------------------------------------------------------
```

```
2744                                  <1>       ;      (AH)= 01H  SET THE ZERO FLAG TO INDICATE IF AN ASCII CHARACTER IS    :
2745                                  <1>       ;                 AVAILABLE TO BE READ FROM THE KEYBOARD BUFFER.             :
2746                                  <1>       ;                 (ZF)= 1 -- NO CODE AVAILABLE                               :
2747                                  <1>       ;                 (ZF)= 0 -- CODE IS AVAILABLE  (AX)= CHARACTER              :
2748                                  <1>       ;                 IF (ZF)= 0, THE NEXT CHARACTER IN THE BUFFER TO BE READ IS :
2749                                  <1>       ;                 IN (AX), AND THE ENTRY REMAINS IN THE BUFFER.              :
2750                                  <1>       ;                 THIS WILL RETURN ONLY PC/PCAT KEYBOARD COMPATIBLE CODES    :
2751                                  <1>       ;-------------------------------------------------------------------------------
2752                                  <1>       ;      (AH)= 02H  RETURN THE CURRENT SHIFT STATUS IN AL REGISTER           :
2753                                  <1>       ;                 THE BIT SETTINGS FOR THIS CODE ARE INDICATED IN THE        :
2754                                  <1>       ;                 EQUATES FOR @KB_FLAG                                      :
2755                                  <1>       ;-------------------------------------------------------------------------------
2756                                  <1>       ;      (AH)= 03H  SET TYPAMATIC RATE AND DELAY                           :
2757                                  <1>       ;            (AL) = 05H                                                  :
2758                                  <1>       ;            (BL) = TYPAMATIC RATE (BITS 5 - 7 MUST BE RESET TO 0)       :
2759                                  <1>       ;                                                                        :
2760                                  <1>       ;                  REGISTER    RATE      REGISTER    RATE                :
2761                                  <1>       ;                   VALUE     SELECTED    VALUE     SELECTED              :
2762                                  <1>       ;                  ------------------------------------------             :
2763                                  <1>       ;                    00H        30.0        10H        7.5                :
2764                                  <1>       ;                    01H        26.7        11H        6.7                :
2765                                  <1>       ;                    02H        24.0        12H        6.0                :
2766                                  <1>       ;                    03H        21.8        13H        5.5                :
2767                                  <1>       ;                    04H        20.0        14H        5.0                :
2768                                  <1>       ;                    05H        18.5        15H        4.6                :
2769                                  <1>       ;                    06H        17.1        16H        4.3                :
2770                                  <1>       ;                    07H        16.0        17H        4.0                :
2771                                  <1>       ;                    08H        15.0        18H        3.7                :
2772                                  <1>       ;                    09H        13.3        19H        3.3                :
2773                                  <1>       ;                    0AH        12.0        1AH        3.0                :
2774                                  <1>       ;                    0BH        10.9        1BH        2.7                :
2775                                  <1>       ;                    0CH        10.0        1CH        2.5                :
2776                                  <1>       ;                    0DH         9.2        1DH        2.3                :
2777                                  <1>       ;                    0EH         8.6        1EH        2.1                :
2778                                  <1>       ;                    0FH         8.0        1FH        2.0                :
2779                                  <1>       ;                                                                        :
2780                                  <1>       ;            (BH) = TYPAMATIC DELAY  (BITS 2 - 7 MUST BE RESET TO 0)     :
2781                                  <1>       ;                                                                       :
2782                                  <1>       ;                     REGISTER     DELAY                                :
2783                                  <1>       ;                      VALUE       VALUE                                :
2784                                  <1>       ;                     ------------------                                :
2785                                  <1>       ;                       00H        250 ms                               :
2786                                  <1>       ;                       01H        500 ms                               :
2787                                  <1>       ;                       02H        750 ms                               :
2788                                  <1>       ;                       03H       1000 ms                               :
2789                                  <1>       ;-------------------------------------------------------------------------------
2790                                  <1>       ;      (AH)= 05H  PLACE ASCII CHARACTER/SCAN CODE COMBINATION IN KEYBOARD  :
2791                                  <1>       ;                 BUFFER AS IF STRUCK FROM KEYBOARD                         :
2792                                  <1>       ;                 ENTRY:  (CL) = ASCII CHARACTER                            :
2793                                  <1>       ;                         (CH) = SCAN CODE                                  :
2794                                  <1>       ;                 EXIT:   (AH) = 00H = SUCCESSFUL OPERATION                 :
2795                                  <1>       ;                         (AL) = 01H = UNSUCCESSFUL - BUFFER FULL           :
2796                                  <1>       ;                 FLAGS:  CARRY IF ERROR                                    :
2797                                  <1>       ;-------------------------------------------------------------------------------
2798                                  <1>       ;      (AH)= 10H  EXTENDED READ INTERFACE FOR THE ENHANCED KEYBOARD,       :
2799                                  <1>       ;                 OTHERWISE SAME AS FUNCTION AH=0                           :
2800                                  <1>       ;-------------------------------------------------------------------------------
2801                                  <1>       ;      (AH)= 11H  EXTENDED ASCII STATUS FOR THE ENHANCED KEYBOARD,         :
2802                                  <1>       ;                 OTHERWISE SAME AS FUNCTION AH=1                           :
2803                                  <1>       ;-------------------------------------------------------------------------------
2804                                  <1>       ;      (AH)= 12H  RETURN THE EXTENDED SHIFT STATUS IN AX REGISTER          :
2805                                  <1>       ;                 AL = BITS FROM KB_FLAG, AH = BITS FOR LEFT AND RIGHT      :
2806                                  <1>       ;                 CTL AND ALT KEYS FROM KB_FLAG_1 AND KB_FLAG_3             :
2807                                  <1>       ; OUTPUT                                                                    :
2808                                  <1>       ;      AS NOTED ABOVE, ONLY (AX) AND FLAGS CHANGED                          :
2809                                  <1>       ;      ALL REGISTERS RETAINED                                               :
2810                                  <1>       ;-------------------------------------------------------------------------------
2811                                  <1>
2812                                  <1> ; 12/04/2021 - TRDOS 386 v2.0.3 (32 bit push/pop)
2813                                  <1> ; 15/01/2017
2814                                  <1> ; 14/01/2017
2815                                  <1> ; 02/01/2017
2816                                  <1> ; 29/05/2016
2817                                  <1> ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
2818                                  <1> int32h:  ; Keyboard BIOS
2819                                  <1>
2820                                  <1> KEYBOARD_IO_1:
2821                                  <1>       ;sti                    ; INTERRUPTS BACK ON
2822                                  <1>       ; 29/05/2016
2823 00000EE3 80642408BE             <1>         and     byte [esp+8], 10111110b ; clear zero flag and cary flag
2824                                  <1>       ;
2825 00000EE8 1E                     <1>       push  ds                ; SAVE CURRENT DS
2826 00000EE9 53                     <1>       push  ebx               ; SAVE BX TEMPORARILY
2827                                  <1>       ;push ecx               ; SAVE CX TEMPORARILY
2828 00000EEA 66BB1000               <1>       mov   bx, KDATA
2829 00000EEE 8EDB                   <1>       mov   ds, bx            ; PUT SEGMENT VALUE OF DATA AREA INTO DS
2830                                  <1>       ; 14/01/2017
2831 00000EF0 8B1C24                 <1>       mov   ebx, [esp]
2832                                  <1>       ;; 15/01/2017
2833                                  <1>       ; 02/01/2017
2834                                  <1>       ;;mov byte [intflg], 32h  ; keyboard interrupt
2835 00000EF3 FB                     <1>       sti
2836                                  <1>       ;
2837 00000EF4 08E4                   <1>       or    ah, ah            ; CHECK FOR (AH)= 00H
2838 00000EF6 745B                   <1>       jz    short _K1         ; ASCII_READ
2839 00000EF8 FECC                   <1>       dec   ah                ; CHECK FOR (AH)= 01H
2840 00000EFA 7474                   <1>       jz    short _K2         ; ASCII_STATUS
2841 00000EFC FECC                   <1>       dec   ah                ; CHECK FOR (AH)= 02H
2842 00000EFE 7440                   <1>       jz    short _K3         ; SHIFT STATUS
2843 00000F00 FECC                   <1>       dec   ah                ; CHECK FOR (AH)= 03H
2844 00000F02 0F8493000000           <1>       jz    _K300             ; SET TYPAMATIC RATE/DELAY
2845 00000F08 80EC02                 <1>       sub   ah, 2             ; CHECK FOR (AH)= 05H
2846 00000F0B 0F84BC000000           <1>       jz    _K500             ; KEYBOARD WRITE
2847                                  <1> _KIO1:
2848 00000F11 80EC0B                 <1>       sub   ah, 11            ; AH =  10H
```

```
2849 00000F14 7431              <1>       jz     short _K1E          ; EXTENDED ASCII READ
2850 00000F16 FECC              <1>       dec    ah                  ; CHECK FOR (AH)= 11H
2851 00000F18 7447              <1>       jz     short _K2E          ; EXTENDED_ASCII_STATUS
2852 00000F1A FECC              <1>       dec    ah                  ; CHECK FOR (AH)= 12H
2853 00000F1C 7404              <1>       jz     short _K3E          ; EXTENDED_SHIFT_STATUS
2854                            <1> _KIO_EXIT:
2855                            <1>       ; 02/01/2017
2856 00000F1E FA                <1>       cli
2857                            <1>       ;;mov  byte [intflg], 0 ;; 15/01/2017
2858                            <1>       ;
2859                            <1>       ;pop   ecx                 ; RECOVER REGISTER
2860 00000F1F 5B                <1>       pop    ebx                 ; RECOVER REGISTER
2861 00000F20 1F                <1>       pop    ds                  ; RECOVER SEGMENT
2862 00000F21 CF                <1>       iretd                      ; INVALID COMMAND, EXIT
2863                            <1>
2864                            <1>       ;----- SHIFT STATUS
2865                            <1> _K3E:                            ; GET THE EXTENDED SHIFT STATUS FLAGS
2866 00000F22 8A25[866A0000]    <1>       mov    ah, [KB_FLAG_1]     ; GET SYSTEM SHIFT KEY STATUS
2867 00000F28 80E404            <1>       and    ah, SYS_SHIFT       ; MASK ALL BUT SYS KEY BIT
2868                            <1>       ;mov   cl, 5               ; SHIFT THEW SYSTEMKEY BIT OVER TO
2869                            <1>       ;shl   ah, cl              ; BIT 7 POSITION
2870 00000F2B C0E405            <1>       shl    ah, 5
2871 00000F2E A0[866A0000]      <1>       mov    al, [KB_FLAG_1]     ; GET SYSTEM SHIFT STATES BACK
2872 00000F33 2473              <1>       and    al, 01110011b       ; ELIMINATE SYS SHIFT, HOLD_STATE AND INS_SHIFT
2873 00000F35 08C4              <1>       or     ah, al              ; MERGE REMAINING BITS INTO AH
2874 00000F37 A0[886A0000]      <1>       mov    al, [KB_FLAG_3]     ; GET RIGHT CTL AND ALT
2875 00000F3C 240C              <1>       and    al, 00001100b       ; ELIMINATE LC_E0 AND LC_E1
2876 00000F3E 08C4              <1>       or     ah, al              ; OR THE SHIFT FLAGS TOGETHER
2877                            <1> _K3:
2878 00000F40 A0[856A0000]      <1>       mov    al, [KB_FLAG]       ; GET THE SHIFT STATUS FLAGS
2879                            <1>       ;jmp   short _KIO_EXIT     ; RETURN TO CALLER
2880 00000F45 EBD7              <1>       jmp    _KIO_EXIT
2881                            <1>
2882                            <1>       ;----- ASCII CHARACTER
2883                            <1> _K1E:
2884 00000F47 E8AE000000        <1>       call   _K1S                ; GET A CHARACTER FROM THE BUFFER (EXTENDED)
2885 00000F4C E821010000        <1>       call   _KIO_E_XLAT         ; ROUTINE TO XLATE FOR EXTENDED CALLS
2886 00000F51 EBCB              <1>       jmp    short _KIO_EXIT     ; GIVE IT TO THE CALLER
2887                            <1> _K1:
2888 00000F53 E8A2000000        <1>       call   _K1S                ; GET A CHARACTER FROM THE BUFFER
2889 00000F58 E820010000        <1>       call   _KIO_S_XLAT         ; ROUTINE TO XLATE FOR STANDARD CALLS
2890 00000F5D 72F4              <1>       jc     short _K1           ; CARRY SET MEANS TROW CODE AWAY
2891                            <1> _K1A:
2892 00000F5F EBBD              <1>       jmp    short _KIO_EXIT     ; RETURN TO CALLER
2893                            <1>
2894                            <1>       ;----- ASCII STATUS
2895                            <1> _K2E:
2896 00000F61 E8DF000000        <1>       call   _K2S                ; TEST FOR CHARACTER IN BUFFER (EXTENDED)
2897 00000F66 7420              <1>       jz     short _K2B          ; RETURN IF BUFFER EMPTY
2898 00000F68 9C                <1>       pushf                      ; SAVE ZF FROM TEST
2899 00000F69 E804010000        <1>       call   _KIO_E_XLAT         ; ROUTINE TO XLATE FOR EXTENDED CALLS
2900 00000F6E EB17              <1>       jmp    short _K2A          ; GIVE IT TO THE CALLER
2901                            <1> _K2:
2902 00000F70 E8D0000000        <1>       call   _K2S                ; TEST FOR CHARACTER IN BUFFER
2903 00000F75 7411              <1>       jz     short _K2B          ; RETURN IF BUFFER EMPTY
2904 00000F77 9C                <1>       pushf                      ; SAVE ZF FROM TEST
2905 00000F78 E800010000        <1>       call   _KIO_S_XLAT         ; ROUTINE TO XLATE FOR STANDARD CALLS
2906 00000F7D 7308              <1>       jnc    short _K2A          ; CARRY CLEAR MEANS PASS VALID CODE
2907 00000F7F 9D                <1>       popf                       ; INVALID CODE FOR THIS TYPE OF CALL
2908 00000F80 E875000000        <1>       call   _K1S                ; THROW THE CHARACTER AWAY
2909 00000F85 EBE9              <1>       jmp    short _K2           ; GO LOOK FOR NEXT CHAR, IF ANY
2910                            <1> _K2A:
2911 00000F87 9D                <1>       popf                       ; RESTORE ZF FROM TEST
2912                            <1> _K2B:
2913                            <1>       ; 02/01/2017
2914 00000F88 FA                <1>       cli
2915                            <1>       ;; mov byte [intflg], 0 ;; 15/01/2017
2916                            <1>       ;
2917                            <1>       ;pop   ecx                 ; RECOVER REGISTER
2918 00000F89 5B                <1>       pop    ebx                 ; RECOVER REGISTER
2919 00000F8A 1F                <1>       pop    ds                  ; RECOVER SEGMENT
2920                            <1>       ; (*) 29/05/2016
2921                            <1>       ; (*) retf 4               ; THROW AWAY (e)FLAGS
2922 00000F8B 7208              <1>       jc     short _k2d
2923 00000F8D 7505              <1>       jnz    short _k2c
2924 00000F8F 804C240840        <1>       or     byte [esp+8], 01000000b   ; set zero flag bit of eflags register
2925                            <1> _k2c:
2926 00000F94 CF                <1>       iretd
2927                            <1> _k2d:
2928                            <1>       ; 29/05/2016 -set carry flag on stack-
2929                            <1>       ; [esp] = EIP
2930                            <1>       ; [esp+4] = CS
2931                            <1>       ; [esp+8] = E-FLAGS
2932 00000F95 804C240801        <1>       or     byte [esp+8], 1  ; set carry bit of eflags register
2933                            <1>       ; [esp+12] = ESP (user)
2934                            <1>       ; [esp+16] = SS (User)
2935 00000F9A CF                <1>       iretd
2936                            <1>
2937                            <1>       ; (*) 29/05/2016 - 'ref 4' intruction causes to stack fault
2938                            <1>       ; (OUTER-PRIVILEGE-LEVEL)
2939                            <1>       ; INTEL 80386 PROGRAMMER'S REFERENCE MANUAL 1986
2940                            <1>       ; // RETF instruction:
2941                            <1>       ;
2942                            <1>       ; IF OperandMode=32 THEN
2943                            <1>       ;    Load CS:EIP from stack;
2944                            <1>       ;    Set CS RPL to CPL;
2945                            <1>       ;    Increment eSP by 8 plus the immediate offset if it exists;
2946                            <1>       ;    Load SS:eSP from stack;
2947                            <1>       ; ELSE (* OperandMode=16 *)
2948                            <1>       ;    Load CS:IP from stack;
2949                            <1>       ;    Set CS RPL to CPL;
2950                            <1>       ;    Increment eSP by 4 plus the immediate offset if it exists;
2951                            <1>       ;    Load SS:eSP from stack;
2952                            <1>       ; FI;
2953                            <1>       ;
```

```
                        <1>       ; //
                        <1>
                        <1>       ;----- SET TYPAMATIC RATE AND DELAY
                        <1> _K300:
00000F9B 3C05           <1>       cmp   al, 5               ; CORRECT FUNCTION CALL?
                        <1>       ;jne  short _KIO_EXIT     ; NO, RETURN
00000F9D 0F857BFFFFFF   <1>       jne   _KIO_EXIT
00000FA3 F6C3E0         <1>       test  bl, 0E0h            ; TEST FOR OUT-OF-RANGE RATE
00000FA6 0F8572FFFFFF   <1>       jnz   _KIO_EXIT           ; RETURN IF SO
00000FAC F6C7FC         <1>       test  BH, 0FCh            ; TEST FOR OUT-OF-RANGE DELAY
00000FAF 0F8569FFFFFF   <1>       jnz   _KIO_EXIT           ; RETURN IF SO
00000FB5 B0F3           <1>       mov   al, KB_TYPA_RD      ; COMMAND FOR TYPAMATIC RATE/DELAY
00000FB7 E87E060000     <1>       call  SND_DATA            ; SEND TO KEYBOARD
                        <1>       ;mov  cx, 5               ; SHIFT COUNT
                        <1>       ;shl  bh, cl              ; SHIFT DELAY OVER
00000FBC C0E705         <1>       shl   bh, 5
00000FBF 88D8           <1>       mov   al, bl              ; PUT IN RATE
00000FC1 08F8           <1>       or    al, bh              ; AND DELAY
00000FC3 E872060000     <1>       call  SND_DATA            ; SEND TO KEYBOARD
00000FC8 E951FFFFFF     <1>       jmp   _KIO_EXIT           ; RETURN TO CALLER
                        <1>
                        <1>       ;----- WRITE TO KEYBOARD BUFFER
                        <1> _K500:
00000FCD 56             <1>       push  esi                 ; SAVE SI (esi)
00000FCE FA             <1>       cli                       ;
00000FCF 8B1D[966A0000] <1>       mov   ebx, [BUFFER_TAIL]  ; GET THE 'IN TO' POINTER TO THE BUFFER
00000FD5 89DE           <1>       mov   esi, ebx            ; SAVE A COPY IN CASE BUFFER NOT FULL
00000FD7 E8D1000000     <1>       call  _K4                 ; BUMP THE POINTER TO SEE IF BUFFER IS FULL
00000FDC 3B1D[926A0000] <1>       cmp   ebx, [BUFFER_HEAD]  ; WILL THE BUFFER OVERRUN IF WE STORE THIS?
00000FE2 740D           <1>       je    short _K502         ; YES - INFORM CALLER OF ERROR
00000FE4 66890E         <1>       mov   [esi], cx           ; NO - PUT ASCII/SCAN CODE INTO BUFFER
00000FE7 891D[966A0000] <1>       mov   [BUFFER_TAIL], ebx  ; ADJUST 'IN TO' POINTER TO REFLECT CHANGE
00000FED 28C0           <1>       sub   al, al              ; TELL CALLER THAT OPERATION WAS SUCCESSFUL
00000FEF EB02           <1>       jmp   short _K504         ; SUB INSTRUCTION ALSO RESETS CARRY FLAG
                        <1> _K502:
00000FF1 B001           <1>       mov   al, 01h             ; BUFFER FULL INDICATION
                        <1> _K504:
00000FF3 FB             <1>       sti
00000FF4 5E             <1>       pop   esi                 ; RECOVER SI (esi)
00000FF5 E924FFFFFF     <1>       jmp   _KIO_EXIT           ; RETURN TO CALLER WITH STATUS IN AL
                        <1>
                        <1>       ;----- READ THE KEY TO FIGURE OUT WHAT TO DO -----
                        <1> _K1S:
00000FFA FA             <1>       cli   ; 03/12/2014
00000FFB 8B1D[926A0000] <1>       mov    ebx, [BUFFER_HEAD] ; GET POINTER TO HEAD OF BUFFER
00001001 3B1D[966A0000] <1>       cmp    ebx, [BUFFER_TAIL] ; TEST END OF BUFFER
                        <1>       ;jne  short _K1U          ; IF ANYTHING IN BUFFER SKIP INTERRUPT
00001007 750F           <1>       jne   short _k1x ; 03/12/2014
                        <1>       ;
                        <1>       ; 03/12/2014
                        <1>       ; 28/08/2014
                        <1>       ; PERFORM OTHER FUNCTION ?? here !
                        <1>       ;; MOV AX, 9002h          ; MOVE IN WAIT CODE & TYPE
                        <1>       ;; INT 15H                ; PERFORM OTHER FUNCTION
                        <1> _K1T:                           ; ASCII READ
00001009 FB             <1>       sti                       ; INTERRUPTS BACK ON DURING LOOP
0000100A 90             <1>       nop                       ; ALLOW AN INTERRUPT TO OCCUR
                        <1> _K1U:
0000100B FA             <1>       cli                       ; INTERRUPTS BACK OFF
0000100C 8B1D[926A0000] <1>       mov   ebx, [BUFFER_HEAD]  ; GET POINTER TO HEAD OF BUFFER
00001012 3B1D[966A0000] <1>       cmp   ebx, [BUFFER_TAIL]  ; TEST END OF BUFFER
                        <1> _k1x:
00001018 53             <1>       push  ebx                 ; SAVE ADDRESS
00001019 9C             <1>       pushf                     ; SAVE FLAGS
0000101A E8CF060000     <1>       call  MAKE_LED            ; GO GET MODE INDICATOR DATA BYTE
0000101F 8A1D[876A0000] <1>       mov   bl, [KB_FLAG_2]     ; GET PREVIOUS BITS
00001025 30C3           <1>       xor   bl, al              ; SEE IF ANY DIFFERENT
00001027 80E307         <1>       and   bl, 07h ; KB_LEDS   ; ISOLATE INDICATOR BITS
0000102A 7406           <1>       jz    short _K1V          ; IF NO CHANGE BYPASS UPDATE
0000102C E869060000     <1>       call  SND_LED1
00001031 FA             <1>       cli                       ; DISABLE INTERRUPTS
                        <1>  K1V:
00001032 9D             <1>       popf                      ; RESTORE FLAGS
00001033 5B             <1>       pop   ebx                 ; RESTORE ADDRESS
00001034 74D3           <1>       je    short _K1T          ; LOOP UNTIL SOMETHING IN BUFFER
                        <1>       ;
00001036 668B03         <1>       mov   ax, [ebx]           ; GET SCAN CODE AND ASCII CODE
00001039 E86F000000     <1>       call  _K4                 ; MOVE POINTER TO NEXT POSITION
0000103E 891D[926A0000] <1>       mov   [BUFFER_HEAD], ebx  ; STORE VALUE IN VARIABLE
00001044 C3             <1>       retn                      ; RETURN
                        <1>
                        <1>       ;----- READ THE KEY TO SEE IF ONE IS PRESENT -----
                        <1> _K2S:
00001045 FA             <1>       cli                       ; INTERRUPTS OFF
00001046 8B1D[926A0000] <1>       mov   ebx, [BUFFER_HEAD]  ; GET HEAD POINTER
0000104C 3B1D[966A0000] <1>       cmp   ebx, [BUFFER_TAIL]  ; IF EQUAL (Z=1) THEN NOTHING THERE
00001052 668B03         <1>       mov    ax, [ebx]
00001055 9C             <1>       pushf                     ; SAVE FLAGS
                        <1>       ;push  ax                 ; SAVE CODE
                        <1>       ; 12/04/2021
00001056 50             <1>       push   eax
00001057 E892060000     <1>       call   MAKE_LED           ; GO GET MODE INDICATOR DATA BYTE
0000105C 8A1D[876A0000] <1>       mov    bl, [KB_FLAG_2]    ; GET PREVIOUS BITS
00001062 30C3           <1>       xor    bl, al             ; SEE IF ANY DIFFERENT
00001064 80E307         <1>       and    bl, 07h ; KB_LEDS  ; ISOLATE INDICATOR BITS
00001067 7405           <1>       jz     short _K2T         ; IF NO CHANGE BYPASS UPDATE
00001069 E815060000     <1>       call   SND_LED            ; GO TURN ON MODE INDICATORS
                        <1> _K2T:
                        <1>       ;pop   ax                 ; RESTORE CODE
                        <1>       ; 12/04/2021
0000106E 58             <1>       pop    eax
0000106F 9D             <1>       popf                      ; RESTORE FLAGS
00001070 FB             <1>       sti                       ; INTERRUPTS BACK ON
00001071 C3             <1>       retn                      ; RETURN
                        <1>
```

```
3059                            <1>       ;----- ROUTINE TO TRANSLATE SCAN CODE PAIRS FOR EXTENDED CALLS -----
3060                            <1> _KIO_E_XLAT:
3061 00001072 3CF0              <1>       cmp    al, 0F0h           ; IS IT ONE OF THE FILL-INs?
3062 00001074 7506              <1>       jne    short _KIO_E_RET   ; NO, PASS IT ON
3063 00001076 08E4              <1>       or     ah, ah             ; AH = 0 IS SPECIAL CASE
3064 00001078 7402              <1>       jz     short _KIO_E_RET    ; PASS THIS ON UNCHANGED
3065 0000107A 30C0              <1>       xor    al, al             ; OTHERWISE SET AL = 0
3066                            <1> _KIO_E_RET:
3067 0000107C C3                <1>       retn                      ; GO BACK
3068                            <1>
3069                            <1>       ;----- ROUTINE TO TRANSLATE SCAN CODE PAIRS FOR STANDARD CALLS -----
3070                            <1> _KIO_S_XLAT:
3071 0000107D 80FCE0            <1>       cmp    ah, 0E0h           ; IS IT KEYPAD ENTER OR / ?
3072 00001080 750F              <1>       jne    short _KIO_S2      ; NO, CONTINUE
3073 00001082 3C0D              <1>       cmp    al, 0Dh            ; KEYPAD ENTER CODE?
3074 00001084 7408              <1>       je     short _KIO_S1      ; YES, MASSAGE A BIT
3075 00001086 3C0A              <1>       cmp    al, 0Ah            ; CTRL KEYPAD ENTER CODE?
3076 00001088 7404              <1>       je     short _KIO_S1      ; YES, MASSAGE THE SAME
3077 0000108A B435              <1>       mov    ah, 35h            ; NO, MUST BE KEYPAD /
3078                            <1> _kio_ret: ; 03/12/2014
3079 0000108C F8                <1>       clc
3080 0000108D C3                <1>       retn
3081                            <1>       ;jmp   short _KIO_USE     ; GIVE TO CALLER
3082                            <1> _KIO_S1:
3083 0000108E B41C              <1>       mov    ah, 1Ch            ; CONVERT TO COMPATIBLE OUTPUT
3084                            <1>       ;jmp   short _KIO_USE     ; GIVE TO CALLER
3085 00001090 C3                <1>       retn
3086                            <1> _KIO_S2:
3087 00001091 80FC84            <1>       cmp    ah, 84h            ; IS IT ONE OF EXTENDED ONES?
3088 00001094 7715              <1>       ja     short _KIO_DIS     ; YES, THROW AWAY AND GET ANOTHER CHAR
3089 00001096 3CF0              <1>       cmp    al, 0F0h           ; IS IT ONE OF THE FILL-INs?
3090 00001098 7506              <1>       jne    short _KIO_S3      ; NO, TRY LAST TEST
3091 0000109A 08E4              <1>       or     ah, ah             ; AH = 0 IS SPECIAL CASE
3092 0000109C 740C              <1>       jz     short _KIO_USE     ; PASS THIS ON UNCHANGED
3093 0000109E EB0B              <1>       jmp    short _KIO_DIS     ; THROW AWAY THE REST
3094                            <1> _KIO_S3:
3095 000010A0 3CE0              <1>       cmp    al, 0E0h           ; IS IT AN EXTENSION OF A PREVIOUS ONE?
3096                            <1>       ;jne   short _KIO_USE     ; NO, MUST BE A STANDARD CODE
3097 000010A2 75E8              <1>       jne    short _kio_ret
3098 000010A4 08E4              <1>       or     ah, ah             ; AH = 0 IS SPECIAL CASE
3099 000010A6 7402              <1>       jz     short _KIO_USE     ; JUMP IF AH = 0
3100 000010A8 30C0              <1>       xor    al, al             ; CONVERT TO COMPATIBLE OUTPUT
3101                            <1>       ;jmp   short _KIO_USE     ; PASS IT ON TO CALLER
3102                            <1> _KIO_USE:
3103                            <1>       ;clc                      ; CLEAR CARRY TO INDICATE GOOD CODE
3104 000010AA C3                <1>       retn                      ; RETURN
3105                            <1> _KIO_DIS:
3106 000010AB F9                <1>       stc                       ; SET CARRY TO INDICATE DISCARD CODE
3107 000010AC C3                <1>       retn                      ; RETURN
3108                            <1>
3109                            <1>       ;----- INCREMENT BUFFER POINTER ROUTINE -----
3110                            <1> _K4:
3111 000010AD 43                <1>       inc   ebx
3112 000010AE 43                <1>       inc    ebx                ; MOVE TO NEXT WORD IN LIST
3113 000010AF 3B1D[8E6A0000]    <1>       cmp   ebx, [BUFFER_END]   ; AT END OF BUFFER?
3114                            <1>       ;jne  short _K5           ; NO, CONTINUE
3115 000010B5 7206              <1>       jb     short _K5
3116 000010B7 8B1D[8A6A0000]    <1>       mov   ebx, [BUFFER_START] ; YES, RESET TO BUFFER BEGINNING
3117                            <1> _K5:
3118 000010BD C3                <1>       retn
3119                            <1>
3120                            <1> ; 20/02/2015
3121                            <1> ; 05/12/2014
3122                            <1> ; 26/08/2014
3123                            <1> ; KEYBOARD (HARDWARE) INTERRUPT -  IRQ LEVEL 1
3124                            <1> ; (INT_09h - Retro UNIX 8086 v1 - U9.ASM, 07/03/2014)
3125                            <1> ;
3126                            <1> ; Derived from "KB_INT_1" procedure of IBM "pc-at"
3127                            <1> ; rombios source code (06/10/1985)
3128                            <1> ; 'keybd.asm', HARDWARE INT 09h - (IRQ Level 1)
3129                            <1>
3130                            <1> ; EQUATES (IBM PC-XT-286 BIOS, 1986, 'POSQEQU.INC')
3131                            <1>
3132                            <1> ;--------- 8042 COMMANDS --------------------------------------------------------
3133                            <1> ENA_KBD      equ    0AEh         ; ENABLE KEYBOARD COMMAND
3134                            <1> DIS_KBD      equ    0ADh         ; DISABLE KEYBOARD COMMAND
3135                            <1> SHUT_CMD     equ    0FEh         ; CAUSE A SHUTDOWN COMMAND
3136                            <1> ;--------- 8042 KEYBOARD INTERFACE AND DIAGNOSTIC CONTROL REGISTERS ------------
3137                            <1> STATUS_PORT  equ    064h         ; 8042 STATUS PORT
3138                            <1> INPT_BUF_FULL equ  00000010b    ; 1 = +INPUT BUFFER FULL
3139                            <1> PORT_A       equ    060h         ; 8042 KEYBOARD SCAN CODE/CONTROL PORT
3140                            <1> ;---------- 8042 KEYBOARD RESPONSE ----------------------------------------------
3141                            <1> KB_ACK       equ    0FAh         ; ACKNOWLEDGE PROM TRANSMISSION
3142                            <1> KB_RESEND    equ    0FEh         ; RESEND REQUEST
3143                            <1> KB_OVER_RUN  equ    0FFh         ; OVER RUN SCAN CODE
3144                            <1> ;---------- KEYBOARD/LED COMMANDS -----------------------------------------------
3145                            <1> KB_ENABLE    equ    0F4h         ; KEYBOARD ENABLE
3146                            <1> LED_CMD      equ    0EDh         ; LED WRITE COMMAND
3147                            <1> KB_TYPA_RD   equ    0F3h         ; TYPAMATIC RATE/DELAY COMMAND
3148                            <1> ;---------- KEYBOARD SCAN CODES -------------------------------------------------
3149                            <1> NUM_KEY      equ    69           ; SCAN CODE FOR      NUMBER LOCK KEY
3150                            <1> SCROLL_KEY   equ    70           ; SCAN CODE FOR      SCROLL LOCK KEY
3151                            <1> ALT_KEY      equ    56           ; SCAN CODE FOR      ALTERNATE SHIFT KEY
3152                            <1> CTL_KEY      equ    29           ; SCAN CODE FOR      CONTROL KEY
3153                            <1> CAPS_KEY     equ    58           ; SCAN CODE FOR      SHIFT LOCK KEY
3154                            <1> DEL_KEY      equ    83           ; SCAN CODE FOR      DELETE KEY
3155                            <1> INS_KEY      equ    82           ; SCAN CODE FOR      INSERT KEY
3156                            <1> LEFT_KEY     equ    42           ; SCAN CODE FOR      LEFT SHIFT
3157                            <1> RIGHT_KEY    equ    54           ; SCAN CODE FOR      RIGHT SHIFT
3158                            <1> SYS_KEY      equ    84           ; SCAN CODE FOR      SYSTEM KEY
3159                            <1> ;---------- ENHANCED KEYBOARD SCAN CODES ----------------------------------------
3160                            <1> ID_1         equ    0ABh         ; 1ST ID CHARACTER FOR KBX
3161                            <1> ID_2         equ    041h         ; 2ND ID CHARACTER FOR KBX
3162                            <1> ID_2A        equ    054h         ; ALTERNATE 2ND ID CHARACTER FOR KBX
3163                            <1> F11_M        equ    87           ; F11 KEY MAKE
```

```
                        <1> F12_M       equ    88           ; F12 KEY MAKE
                        <1> MC_E0       equ    224          ; GENERAL MARKER CODE
                        <1> MC_E1       equ    225          ; PAUSE KEY MARKER CODE
                        <1> ;---------- FLAG EQUATES WITHIN @KB_FLAG-----------------------------------------
                        <1> RIGHT_SHIFT  equ    00000001b    ; RIGHT SHIFT KEY DEPRESSED
                        <1> LEFT_SHIFT   equ    00000010b    ; LEFT SHIFT KEY DEPRESSED
                        <1> CTL_SHIFT    equ    00000100b    ; CONTROL SHIFT KEY DEPRESSED
                        <1> ALT_SHIFT    equ    00001000b    ; ALTERNATE SHIFT KEY DEPRESSED
                        <1> SCROLL_STATE equ    00010000b    ; SCROLL LOCK STATE IS ACTIVE
                        <1> NUM_STATE    equ    00100000b    ; NUM LOCK STATE IS ACTIVE
                        <1> CAPS_STATE   equ    01000000b    ; CAPS LOCK STATE IS ACTIVE
                        <1> INS_STATE    equ    10000000b    ; INSERT STATE IS ACTIVE
                        <1> ;---------- FLAG EQUATES WITHIN @KB_FLAG_1 -------------------------------------
                        <1> L_CTL_SHIFT  equ    00000001b    ; LEFT CTL KEY DOWN
                        <1> L_ALT_SHIFT  equ    00000010b    ; LEFT ALT KEY DOWN
                        <1> SYS_SHIFT    equ    00000100b    ; SYSTEM KEY DEPRESSED AND HELD
                        <1> HOLD_STATE   equ    00001000b    ; SUSPEND KEY HAS BEEN TOGGLED
                        <1> SCROLL_SHIFT equ    00010000b    ; SCROLL LOCK KEY IS DEPRESSED
                        <1> NUM_SHIFT    equ    00100000b    ; NUM LOCK KEY IS DEPRESSED
                        <1> CAPS_SHIFT   equ    01000000b    ; CAPS LOCK KEY IS DEPRE55ED
                        <1> INS_SHIFT    equ    10000000b    ; INSERT KEY IS DEPRESSED
                        <1> ;---------- FLAGS EQUATES WITHIN @KB_FLAG_2 -------------------------------------
                        <1> KB_LEDS      equ    00000111b    ; KEYBOARD LED STATE BITS
                        <1> ;            equ    00000001b    ; SCROLL LOCK INDICATOR
                        <1> ;            equ    00000010b    ; NUM LOCK INDICATOR
                        <1> ;            equ    00000100b    ; CAPS LOCK INDICATOR
                        <1> ;            equ    00001000b    ; RESERVED (MUST BE ZERO)
                        <1> KB_FA        equ    00010000b    ; ACKNOWLEDGMENT RECEIVED
                        <1> KB_FE        equ    00100000b    ; RESEND RECEIVED FLAG
                        <1> KB_PR_LED    equ    01000000b    ; MODE INDICATOR UPDATE
                        <1> KB_ERR       equ    10000000b    ; KEYBOARD TRANSMIT ERROR FLAG
                        <1> ;---------- FLAGS EQUATES WITHIN @KB_FLAG_3 -------------------------------------
                        <1> LC_E1        equ    00000001b    ; LAST CODE WAS THE E1 HIDDEN CODE
                        <1> LC_E0        equ    00000010b    ; LAST CODE WAS THE E0 HIDDEN CODE
                        <1> R_CTL_SHIFT  equ    00000100b    ; RIGHT CTL KEY DOWN
                        <1> R_ALT_SHIFT  equ    00001000b    ; RIGHT ALT KEY DOWN
                        <1> GRAPH_ON     equ    00001000b    ; ALT GRAPHICS KEY DOWN (WT ONLY)
                        <1> KBX          equ    00010000b    ; ENHANCED KEYBOARD INSTALLED
                        <1> SET_NUM_LK   equ    00100000b    ; FORCE NUM LOCK IF READ ID AND KBX
                        <1> LC_AB        equ    01000000b    ; LAST CHARACTER WAS FIRST ID CHARACTER
                        <1> RD_ID        equ    10000000b    ; DOING A READ ID (MUST BE BIT0)
                        <1> ;
                        <1> ;----------- INTERRUPT EQUATES ---------------------------------------------
                        <1> EOI          equ    020h         ; END OF INTERRUPT COMMAND TO 8259
                        <1> INTA00       equ    020h         ; 8259 PORT
                        <1>
                        <1>
                        <1> kb_int:
                        <1>
                        <1> ; 12/04/2021 - TRDOS 386 v2.0.3 (32 bit push/pop)
                        <1> ; 17/10/2015 ('ctrlbrk')
                        <1> ; 05/12/2014
                        <1> ; 04/12/2014 (derived from pc-xt-286 bios source code -1986-)
                        <1> ; 26/08/2014
                        <1> ;
                        <1> ; 03/06/86  KEYBOARD BIOS
                        <1> ;
                        <1> ;--- HARDWARE INT 09H -- (IRQ LEVEL 1) ----------------------------------------
                        <1> ;                                                                            ;
                        <1> ;     KEYBOARD INTERRUPT ROUTINE                                             ;
                        <1> ;                                                                            ;
                        <1> ;--------------------------------------------------------------------------------
                        <1>
                        <1> KB_INT_1:
000010BE FB             <1>       sti                       ; ENABLE INTERRUPTS
                        <1>       ;push  ebp
000010BF 50             <1>       push   eax
000010C0 53             <1>       push   ebx
000010C1 51             <1>       push   ecx
000010C2 52             <1>       push   edx
000010C3 56             <1>       push   esi
000010C4 57             <1>       push   edi
000010C5 1E             <1>       push   ds
000010C6 06             <1>       push   es
000010C7 FC             <1>       cld                       ; FORWARD DIRECTION
000010C8 66B81000       <1>       mov    ax, KDATA
000010CC 8ED8           <1>       mov    ds, ax
000010CE 8EC0           <1>       mov    es, ax
                        <1>       ;
                        <1>       ;-----WAIT FOR KEYBOARD DISABLE COMMAND TO BE ACCEPTED
000010D0 B0AD           <1>       mov    al, DIS_KBD        ; DISABLE THE KEYBOARD COMMAND
000010D2 E851050000     <1>       call   SHIP_IT                 ; EXECUTE DISABLE
000010D7 FA             <1>       cli                       ; DISABLE INTERRUPTS
000010D8 B900000100     <1>       mov    ecx, 10000h        ; SET MAXIMUM TIMEOUT
                        <1> KB_INT_01:
000010DD E464           <1>       in     al, STATUS_PORT           ; READ ADAPTER STATUS
000010DF A802           <1>       test   al, INPT_BUF_FULL  ; CHECK INPUT BUFFER FULL STATUS BIT
000010E1 E0FA           <1>       loopnz KB_INT_01          ; WAIT FOR COMMAND TO BE ACCEPTED
                        <1>       ;
                        <1>       ;-----READ CHARACTER FROM KEYBOARD INTERFACE
000010E3 E460           <1>       in     al, PORT_A         ; READ IN THE CHARACTER
                        <1>       ;
                        <1>       ;-----SYSTEM HOOK INT 15H - FUNCTION 4FH (ON HARDWARE INT LEVEL 9H)
                        <1>       ;MOV   AH, 04FH           ; SYSTEM INTERCEPT - KEY CODE FUNCTION
                        <1>       ;STC                      ; SET CY=1 (IN CASE OF IRET)
                        <1>       ;INT   15H                ; CASETTE CALL (AL)=KEY SCAN CODE
                        <1>       ;                         ; RETURNS CY=1 FOR INVALID FUNCTION
                        <1>       ;JC    KB_INT_02          ; CONTINUE IF CARRY FLAG SET ((AL)=CODE)
                        <1>       ;JMP   K26                ; EXIT IF SYSTEM HANDLES SCAN CODE
                        <1>       ;                         ; EXÝT HANDLES HARDWARE EOI AND ENABLE
                        <1>       ;
                        <1>       ;-----CHECK FOR A RESEND COMMAND TO KEYBOARD
                        <1> KB_INT_02:                      ;        (AL)= SCAN CODE
000010E5 FB             <1>       sti                       ; ENABLE INTERRUPTS AGAIN
000010E6 3CFE           <1>       cmp    al, KB_RESEND      ; IS THE INPUT A RESEND
```

```
3269 000010E8 740E                <1>       je    short KB_INT_4      ; GO IF RESEND
3270                              <1>       ;
3271                              <1>       ;----- CHECK FOR RESPONSE TO A COMMAND TO KEYBOARD
3272 000010EA 3CFA                <1>       cmp   al, KB_ACK          ; IS THE INPUT AN ACKNOWLEDGE
3273 000010EC 7514                <1>       jne   short KB_INT_2      ; GO IF NOT
3274                              <1>       ;
3275                              <1>       ;----- A COMMAND TO THE KEYBOARD WAS ISSUED
3276 000010EE FA                  <1>       cli                       ; DISABLE INTERRUPTS
3277 000010EF 800D[876A0000]10    <1>       or    byte [KB_FLAG_2], KB_FA ; INDICATE ACK RECEIVED
3278                              <1>       ;jmp  K26                 ; RETURN IF NOT ACK RETURNED FOR DATA)
3279                              <1>       ; 12/04/2021
3280 000010F6 EB76                <1>       jmp   short ID_EX  ; K26
3281                              <1>       ;
3282                              <1>       ;----- RESEND THE LAST BYTE
3283                              <1> KB_INT_4:
3284 000010F8 FA                  <1>       cli                       ; DISABLE INTERRUPTS
3285 000010F9 800D[876A0000]20    <1>       or    byte [KB_FLAG_2], KB_FE ; INDICATE RESEND RECEIVED
3286                              <1>       ;jmp  K26                 ; RETURN IF NOT ACK RETURNED FOR DATA)
3287                              <1>       ; 12/04/2021
3288 00001100 EB6C                <1>       jmp   short ID_EX  ; K26
3289                              <1>       ;
3290                              <1> ;-----      UPDATE MODE INDICATORS IF CHANGE IN STATE
3291                              <1> KB_INT_2:
3292                              <1>       ;push  ax                 ; SAVE DATA IN
3293                              <1>       ; 12/04/2021
3294 00001102 50                  <1>       push   eax
3295 00001103 E8E6050000          <1>       call   MAKE_LED           ; GO GET MODE INDICATOR DATA BYTE
3296 00001108 8A1D[876A0000]      <1>       mov    bl, [KB_FLAG_2]    ; GET PREVIOUS BITS
3297 0000110E 30C3                <1>       xor    bl, al             ; SEE IF ANY DIFFERENT
3298 00001110 80E307              <1>       and    bl, KB_LEDS        ; ISOLATE INDICATOR BITS
3299 00001113 7405                <1>       jz     short UP0          ; IF NO CHANGE BYPASS UPDATE
3300 00001115 E869050000          <1>       call   SND_LED                  ; GO TURN ON MODE INDICATORS
3301                              <1> UP0:
3302                              <1>       ;pop   ax                 ; RESTORE DATA IN
3303                              <1>       ; 12/04/2021
3304 0000111A 58                  <1>       pop    eax
3305                              <1> ;--------------------------------------------------------------------------
3306                              <1> ;     START OF KEY PROCESSING                                            ;
3307                              <1> ;--------------------------------------------------------------------------
3308 0000111B 88C4                <1>       mov    ah, al             ; SAVE SCAN CODE IN AH ALSO
3309                              <1>       ;
3310                              <1>       ;----- TEST FOR OVERRUN SCAN CODE FROM KEYBOARD
3311 0000111D 3CFF                <1>       cmp    al, KB_OVER_RUN    ; IS THIS AN OVERRUN CHAR
3312                              <1>       ;je    K62                ; BUFFER_FULL_BEEP
3313                              <1>       ; 12/04/2021
3314 0000111F 7505                <1>       jne    short K16
3315 00001121 E9EE040000          <1>       jmp    K62
3316                              <1> K16:
3317 00001126 8A3D[886A0000]      <1>       mov    bh, [KB_FLAG_3]    ; LOAD FLAGS FOR TESTING
3318                              <1>       ;
3319                              <1>       ;----- TEST TO SEE IF A READ_ID IS IN PROGRESS
3320 0000112C F6C7C0              <1>       test   bh, RD_ID+LC_AB    ; ARE WE DOING A READ ID?
3321 0000112F 7442                <1>       jz     short NOT_ID       ; CONTINUE IF NOT
3322 00001131 7914                <1>       jns    short TST_ID_2            ; IS THE RD_ID FLAG ON?
3323 00001133 3CAB                <1>       cmp    al, ID_1           ; IS THIS THE 1ST ID CHARACTER?
3324 00001135 7507                <1>       jne    short RST_RD_ID
3325 00001137 800D[886A0000]40    <1>       or     byte [KB_FLAG_3], LC_AB ; INDICATE 1ST ID WAS OK
3326                              <1> RST_RD_ID:
3327 0000113E 8025[886A0000]7F    <1>       and    byte [KB_FLAG_3], ~RD_ID ; RESET THE READ ID FLAG
3328 00001145 EB27                <1>       jmp    short ID_EX        ; AND EXIT
3329                              <1>       ; 12/04/2021
3330                              <1>       ;jmp   K26
3331                              <1>       ;
3332                              <1> TST_ID_2:
3333 00001147 8025[886A0000]BF    <1>       and    byte [KB_FLAG_3], ~LC_AB ; RESET FLAG
3334 0000114E 3C54                <1>       cmp    al, ID_2A          ; IS THIS THE 2ND ID CHARACTER?
3335 00001150 7415                <1>       je     short KX_BIT       ; JUMP IF SO
3336 00001152 3C41                <1>       cmp    al, ID_2           ; IS THIS THE 2ND ID CHARACTER?
3337 00001154 7518                <1>       jne    short ID_EX        ; LEAVE IF NOT
3338                              <1>       ; 12/04/2021
3339                              <1>       ;jne   K26
3340                              <1>       ;
3341                              <1>       ;----- A READ ID SAID THAT IT WAS ENHANCED KEYBOARD
3342 00001156 F6C720              <1>       test   bh, SET_NUM_LK     ; SHOULD WE SET NUM LOCK?
3343 00001159 740C                <1>       jz    short KX_BIT        ; EXIT IF NOT
3344 0000115B 800D[856A0000]20    <1>       or     byte [KB_FLAG], NUM_STATE ; FORCE NUM LOCK ON
3345 00001162 E81C050000          <1>       call   SND_LED            ; GO SET THE NUM LOCK INDICATOR
3346                              <1> KX_BIT:
3347 00001167 800D[886A0000]10    <1>       or     byte [KB_FLAG_3], KBX ; INDICATE ENHANCED KEYBOARD WAS FOUND
3348 0000116E E9CF010000          <1> ID_EX: jmp     K26              ; EXIT
3349                              <1>       ;
3350                              <1> NOT_ID:
3351 00001173 3CE0                <1>       cmp    al, MC_E0          ; IS THIS THE GENERAL MARKER CODE?
3352 00001175 7509                <1>       jne    short TEST_E1
3353 00001177 800D[886A0000]12    <1>       or     byte [KB_FLAG_3], LC_E0+KBX ; SET FLAG BIT, SET KBX, AND
3354 0000117E EB0B                <1>       jmp    short EXIT         ; THROW AWAY THIS CODE
3355                              <1>       ; 12/04/2021
3356                              <1>       ;jmp   K26A
3357                              <1> TEST_E1:
3358 00001180 3CE1                <1>       cmp    al, MC_E1          ; IS THIS THE PAUSE KEY?
3359 00001182 750C                <1>       jne    short NOT_HC
3360 00001184 800D[886A0000]11    <1>       or     byte [KB_FLAG_3], LC_E1+KBX ; SET FLAG BIT, SET KBX, AND
3361 0000118B E9B9010000          <1> EXIT: jmp    K26A               ; THROW AWAY THIS CODE
3362                              <1>       ;
3363                              <1> NOT_HC:
3364 00001190 247F                <1>       and    al, 07Fh           ; TURN OFF THE BREAK BIT
3365 00001192 F6C702              <1>       test   bh, LC_E0          ; LAST CODE THE E0 MARKER CODE
3366 00001195 7410                <1>       jz     short NOT_LC_E0          ; JUMP IF NOT
3367                              <1>       ;
3368 00001197 BF[72690000]        <1>       mov    edi, _K6+6         ; IS THIS A SHIFT KEY?
3369 0000119C AE                  <1>       scasb
3370                              <1>       ;je    K26 ; K16B             ; YES, THROW AWAY & RESET FLAG
3371                              <1>       ; 12/04/2021
3372 0000119D 745F                <1>       je     short K16B ; K26
3373 0000119F AE                  <1>       scasb
```

```
3374 000011A0 7571              <1>       jne    short K16A          ; NO, CONTINUE KEY PROCESSING
3375                            <1>       ;jmp   short K16B          ; YES, THROW AWAY & RESET FLAG
3376 000011A2 E99B010000        <1>       jmp    K26
3377                            <1>       ;
3378                            <1> NOT_LC_E0:
3379 000011A7 F6C701            <1>       test   bh, LC_E1           ; LAST CODE THE E1 MARKER CODE?
3380 000011AA 7429              <1>       jz     short T_SYS_KEY           ; JUMP IF NOT
3381 000011AC B904000000        <1>       mov    ecx, 4              ; LENGHT OF SEARCH
3382 000011B1 BF[70690000]      <1>       mov    edi, _K6+4          ; IS THIS AN ALT, CTL, OR SHIFT?
3383 000011B6 F2AE              <1>       repne  scasb               ; CHECK IT
3384 000011B8 74D1              <1>       je     short EXIT          ; THROW AWAY IF SO
3385                            <1>       ; 12/04/2021
3386                            <1>       ;je    K26A
3387                            <1>       ;
3388 000011BA 3C45              <1>       cmp    al, NUM_KEY         ; IS IT THE PAUSE KEY?
3389 000011BC 7540              <1>       jne    short K16B          ; NO, THROW AWAY & RESET FLAG
3390                            <1>       ; 12/04/2021
3391                            <1>       ;jne   K26
3392 000011BE F6C480            <1>       test   ah, 80h                   ; YES, IS IT THE BREAK OF THE KEY?
3393                            <1>       ;jnz   short K16B          ; YES, THROW THIS AWAY, TOO
3394 000011C1 0F857B010000      <1>       jnz    K26
3395                            <1>         ; 20/02/2015
3396 000011C7 F605[866A0000]08  <1>       test   byte [KB_FLAG_1],HOLD_STATE ; NO, ARE WE PAUSED ALREADY?
3397 000011CE 752E              <1>       jnz    short K16B          ; YES, THROW AWAY
3398                            <1>       ; 12/04/2021
3399                            <1>       ;jnz   K26
3400 000011D0 E9B1020000        <1>       jmp     K39P               ; NO, THIS IS THE REAL PAUSE STATE
3401                            <1>       ;
3402                            <1>       ;----- TEST FOR SYSTEM KEY
3403                            <1> T_SYS_KEY:
3404 000011D5 3C54              <1>       cmp    al, SYS_KEY         ; IS IT THE SYSTEM KEY?
3405 000011D7 753A              <1>       jnz    short K16A          ; CONTINUE IF NOT
3406                            <1>       ;
3407 000011D9 F6C480            <1>       test   ah, 80h             ; CHECK IF THIS A BREAK CODE
3408 000011DC 7525              <1>       jnz    short K16C          ; DO NOT TOUCH SYSTEM INDICATOR IF TRUE
3409                            <1>       ;
3410 000011DE F605[866A0000]04  <1>       test   byte [KB_FLAG_1], SYS_SHIFT ; SEE IF IN SYSTEM KEY HELD DOWN
3411 000011E5 7517              <1>       jnz    short K16B          ; IF YES, DO NOT PROCESS SYSTEM INDICATOR
3412                            <1>       ; 12/04/2021
3413                            <1>       ;jnz   K26
3414                            <1>       ;
3415 000011E7 800D[866A0000]04  <1>       or     byte [KB_FLAG_1], SYS_SHIFT ; INDICATE SYSTEM KEY DEPRESSED
3416 000011EE B020              <1>       mov    al, EOI             ; END OF INTERRUPT COMMAND
3417 000011F0 E620              <1>       out    20h, al ;out INTA00, al ; SEND COMMAND TO INTERRUPT CONTROL PORT
3418                            <1>                                  ; INTERRUPT-RETURN-NO-EOI
3419 000011F2 B0AE              <1>       mov    al, ENA_KBD         ; INSURE KEYBOARD IS ENABLED
3420 000011F4 E82F040000        <1>       call   SHIP_IT                   ; EXECUTE ENABLE
3421                            <1>       ; !!! SYSREQ !!! function/system call (INTERRUPT) must be here !!!
3422                            <1>       ;MOV   AL, 8500H           ; FUNCTION VALUE FOR MAKE OF SYSTEM KEY
3423                            <1>       ;STI                       ; MAKE SURE INTERRUPTS ENABLED
3424                            <1>       ;INT   15H                 ; USER INTERRUPT
3425 000011F9 E957010000        <1>       jmp   K27A                 ; END PROCESSING
3426                            <1>       ;
3427 000011FE E93F010000        <1> K16B: jmp    K26                 ; IGNORE SYSTEM KEY
3428                            <1>       ;
3429                            <1> K16C:
3430 00001203 8025[866A0000]FB  <1>       and    byte [KB_FLAG_1], ~SYS_SHIFT ; TURN OFF SHIFT KEY HELD DOWN
3431 0000120A B020              <1>       mov    al, EOI             ; END OF INTERRUPT COMMAND
3432 0000120C E620              <1>       out    20h, al ;out INTA00, al ; SEND COMMAND TO INTERRUPT CONTROL PORT
3433                            <1>                                  ; INTERRUPT-RETURN-NO-EOI
3434                            <1>       ;MOV   AL, ENA_KBD         ; INSURE KEYBOARD IS ENABLED
3435                            <1>       ;CALL  SHIP_IT             ; EXECUTE ENABLE
3436                            <1>       ;
3437                            <1>       ;MOV   AX, 8501H           ; FUNCTION VALUE FOR BREAK OF SYSTEM KEY
3438                            <1>       ;STI                       ; MAKE SURE INTERRUPTS ENABLED
3439                            <1>       ;INT   15H                 ; USER INTERRUPT
3440                            <1>       ;JMP   K27A                ; INGONRE SYSTEM KEY
3441                            <1>       ;
3442 0000120E E93B010000        <1>       jmp     K27                ; IGNORE SYSTEM KEY
3443                            <1>       ;
3444                            <1>       ;----- TEST FOR SHIFT KEYS
3445                            <1> K16A:
3446 00001213 8A1D[856A0000]    <1>       mov    bl, [KB_FLAG]       ; PUT STATE FLAGS IN BL
3447 00001219 BF[6C690000]      <1>       mov    edi, _K6            ; SHIFT KEY TABLE offset
3448 0000121E B908000000        <1>       mov    ecx, _K6L           ; LENGTH
3449 00001223 F2AE              <1>       repne  scasb               ; LOOK THROUGH THE TABLE FOR A MATCH
3450 00001225 88E0              <1>       mov    al, ah              ; RECOVER SCAN CODE
3451                            <1>       ;jne   K25                 ; IF NO MATCH, THEN SHIFT NOT FOUND
3452                            <1>       ; 12/04/2021
3453 00001227 7405              <1>       je     short K17
3454 00001229 E9FC000000        <1>       jmp    K25
3455                            <1>       ;
3456                            <1>       ;------      SHIFT KEY FOUND
3457                            <1> K17:
3458 0000122E 81EF[6D690000]    <1>       sub    edi, _K6+1          ; ADJUST PTR TO SCAN CODE MATCH
3459 00001234 8AA7[74690000]    <1>       mov    ah, [edi+_K7]       ; GET MASK INTO AH
3460 0000123A B102              <1>       mov    cl, 2               ; SETUP COUNT FOR FLAG SHIFTS
3461 0000123C A880              <1>       test   al, 80h             ; TEST FOR BREAK KEY
3462                            <1>       ;jnz   short K23           ; JUMP OF BREAK
3463                            <1>       ; 12/04/2021
3464 0000123E 7405              <1>       jz     short K17C
3465 00001240 E981000000        <1>       jmp    K23
3466                            <1>       ;
3467                            <1>       ;----- SHIFT MAKE FOUND, DETERMINE SET OR TOGGLE
3468                            <1> K17C:
3469 00001245 80FC10            <1>       cmp    ah, SCROLL_SHIFT
3470 00001248 7324              <1>       jae    short K18           ; IF SCROLL SHIFT OR ABOVE, TOGGLE KEY
3471                            <1>       ;
3472                            <1>       ;----- PLAIN SHIFT KEY, SET SHIFT ON
3473 0000124A 0825[856A0000]    <1>       or     [KB_FLAG], ah       ; TURN ON SHIFT BIT
3474 00001250 A80C              <1>       test   al, CTL_SHIFT+ALT_SHIFT ; IS IT ALT OR CTRL?
3475                            <1>       ;;jnz  short K17D          ; YES, MORE FLAGS TO SET
3476                            <1>       ;jz    K26                 ; NO, INTERRUPT RETURN
3477                            <1>       ; 12/04/2021
3478 00001252 7415              <1>       jz     short k17f
```

```
3479                                    <1> K17D:
3480 00001254 F6C702                    <1>         test   bh, LC_E0           ; IS THIS ONE OF NEW KEYS?
3481 00001257 7408                      <1>         jz     short K17E          ; NO, JUMP
3482 00001259 0825[886A0000]            <1>         or     [KB_FLAG_3], ah     ; SET BITS FOR RIGHT CTRL, ALT
3483                                    <1>         ;jmp   K26                 ; INTERRUPT RETURN
3484                                    <1>         ; 12/04/2021
3485 0000125F EB08                      <1>         jmp    short k17f
3486                                    <1> K17E:
3487 00001261 D2EC                      <1>         shr    ah, cl              ; MOVE FLAG BITS TWO POSITIONS
3488 00001263 0825[866A0000]            <1>         or     [KB_FLAG_1], ah     ; SET BITS FOR LEFT CTRL, ALT
3489                                    <1> k17f: ; 12/04/2021
3490 00001269 E9D4000000                <1>         jmp    K26
3491                                    <1>         ;
3492                                    <1>         ;----- TOGGLED SHIFT KEY, TEST FOR 1ST MAKE OR NOT
3493                                    <1> K18:                               ; SHIFT-TOGGLE
3494 0000126E F6C304                    <1>         test   bl, CTL_SHIFT       ; CHECK CTL SHIFT STATE
3495 00001271 7402                      <1>         jz     short K18A          ; JUMP IF NOT CTL STATE
3496                                    <1>         ;jnz   K25                 ; JUMP IF CTL STATE
3497                                    <1>         ; 12/04/2021
3498 00001273 EB1C                      <1>         jmp    short 20a ; K25
3499                                    <1> K18A:
3500 00001275 3C52                      <1>         cmp    al, INS_KEY         ; CHECK FOR INSERT KEY
3501 00001277 7522                      <1>         jne    short K22           ; JUMP IF NOT INSERT KEY
3502 00001279 F6C308                    <1>         test   bl, ALT_SHIFT       ; CHECK FOR ALTERNATE SHIFT
3503 0000127C 7402                      <1>         jz     short K18B          ; JUMP IF NOT ALTERNATE SHIFT
3504                                    <1>         ;jnz   K25                 ; JUMP IF ALTERNATE SHIFT
3505                                    <1>         ; 12/04/2021
3506 0000127E EB11                      <1>         jmp    short k20a ; K25
3507                                    <1> K18B:
3508 00001280 F6C702                    <1>         test   bh, LC_E0 ;20/02/2015 ; IS THIS NEW INSERT KEY?
3509 00001283 7516                      <1>         jnz    short K22           ; YES, THIS ONE'S NEVER A '0'
3510                                    <1> K19:
3511 00001285 F6C320                    <1>         test   bl, NUM_STATE       ; CHECK FOR BASE STATE
3512 00001288 750C                      <1>         jnz    short K21           ; JUMP IF NUM LOCK IS ON
3513 0000128A F6C303                    <1>         test   bl, LEFT_SHIFT+RIGHT_SHIFT ; TEST FOR SHIFT STATE
3514 0000128D 740C                      <1>         jz     short K22           ; JUMP IF BASE STATE
3515                                    <1> K20:                               ; NUMERIC ZERO, NOT INSERT KEY
3516 0000128F 88C4                      <1>         mov    ah, al              ; PUT SCAN CODE BACK IN AH
3517                                    <1> k20a: ; 12/04/2021
3518 00001291 E994000000                <1>       jmp    K25                 ; NUMERAL '0', STNDRD. PROCESSING
3519                                    <1> K21:                               ; MIGHT BE NUMERIC
3520 00001296 F6C303                    <1>         test   bl, LEFT_SHIFT+RIGHT_SHIFT
3521 00001299 74F4                      <1>         jz     short K20           ; IS NUMERIC, STD. PROC.
3522                                    <1>         ;
3523                                    <1> K22:                               ; SHIFT TOGGLE KEY HIT; PROCESS IT
3524 0000129B 8425[866A0000]            <1>         test   ah, [KB_FLAG_1]     ; IS KEY ALREADY DEPRESSED
3525                                    <1>         ;jnz   short K26           ; JUMP IF KEY ALREADY DEPRESSED
3526                                    <1>         ; 12/04/2021
3527 000012A1 75C6                      <1>         jnz    short k17f ; K26
3528                                    <1> K22A:
3529 000012A3 0825[866A0000]            <1>         or     [KB_FLAG_1], ah     ; INDICATE THAT THE KEY IS DEPRESSED
3530 000012A9 3025[856A0000]            <1>         xor    [KB_FLAG], ah       ; TOGGLE THE SHIFT STATE
3531                                    <1>         ;
3532                                    <1>         ;----- TOGGLE LED IF CAPS, NUM  OR SCROLL KEY DEPRESSED
3533 000012AF F6C470                    <1>         test   ah, CAPS_SHIFT+NUM_SHIFT+SCROLL_SHIFT ; SHIFT TOGGLE?
3534 000012B2 7407                      <1>         jz     short K22B          ; GO IF NOT
3535                                    <1>         ;
3536                                    <1>         ; 12/04/2021 (32 bit push/pop)
3537 000012B4 50                        <1>         push   eax ; push ax       ; SAVE SCAN CODE AND SHIFT MASK
3538 000012B5 E8C9030000                <1>         call   SND_LED                  ; GO TURN MODE INDICATORS ON
3539 000012BA 58                        <1>         pop    eax ; pop ax        ; RESTORE SCAN CODE
3540                                    <1> K22B:
3541 000012BB 3C52                      <1>         cmp    al, INS_KEY         ; TEST FOR 1ST MAKE OF INSERT KEY
3542                                    <1>         ;jne   short K26           ; JUMP IF NOT INSERT KEY
3543                                    <1>         ; 12/04/2021
3544 000012BD 75AA                      <1>         jne    short k17f ; K26
3545 000012BF 88C4                      <1>         mov    ah, al               ; SCAN CODE IN BOTH HALVES OF AX
3546 000012C1 E999000000                <1>         jmp    K28                  ; FLAGS UPDATED, PROC. FOR BUFFER
3547                                    <1>         ;
3548                                    <1>         ;----- BREAK SHIFT FOUND
3549                                    <1> K23:                               ; BREAK-SHIFT-FOUND
3550 000012C6 80FC10                    <1>         cmp    ah, SCROLL_SHIFT    ; IS THIS A TOGGLE KEY
3551 000012C9 F6D4                      <1>         not    ah                  ; INVERT MASK
3552 000012CB 7355                      <1>         jae    short K24           ; YES, HANDLE BREAK TOGGLE
3553 000012CD 2025[856A0000]            <1>         and    [KB_FLAG], ah       ; TURN OFF SHIFT BIT
3554 000012D3 80FCFB                    <1>         cmp    ah, ~CTL_SHIFT           ; IS THIS ALT OR CTL?
3555 000012D6 7730                      <1>         ja     short K23D          ; NO, ALL DONE
3556                                    <1>         ;
3557 000012D8 F6C702                    <1>         test   bh, LC_E0           ; 2ND ALT OR CTL?
3558 000012DB 7408                      <1>         jz     short K23A          ; NO, HANSLE NORMALLY
3559 000012DD 2025[886A0000]            <1>         and    [KB_FLAG_3], ah     ; RESET BIT FOR RIGHT ALT OR CTL
3560 000012E3 EB08                      <1>         jmp    short K23B          ; CONTINUE
3561                                    <1> K23A:
3562 000012E5 D2FC                      <1>         sar    ah, cl              ; MOVE THE MASK BIT TWO POSITIONS
3563 000012E7 2025[866A0000]            <1>         and    [KB_FLAG_1], ah     ; RESET BIT FOR LEFT ALT AND CTL
3564                                    <1> K23B:
3565 000012ED 88C4                      <1>         mov    ah, al              ; SAVE SCAN CODE
3566 000012EF A0[886A0000]              <1>         mov    al, [KB_FLAG_3]     ; GET RIGHT ALT & CTRL FLAGS
3567 000012F4 D2E8                      <1>         shr    al, cl              ; MOVE TO BITS 1 & 0
3568 000012F6 0A05[866A0000]            <1>         or     al, [KB_FLAG_1]     ; PUT IN LEFT ALÞT & CTL FLAGS
3569 000012FC D2E0                      <1>         shl    al, cl              ; MOVE BACK TO BITS 3 & 2
3570 000012FE 240C                      <1>         and    al, ALT_SHIFT+CTL_SHIFT ; FILTER OUT OTHER GARBAGE
3571 00001300 0805[856A0000]            <1>         or     [KB_FLAG], al       ; PUT RESULT IN THE REAL FLAGS
3572 00001306 88E0                      <1>         mov    al, ah
3573                                    <1> K23D:
3574 00001308 3CB8                      <1>         cmp    al, ALT_KEY+80h     ; IS THIS ALTERNATE SHIFT RELEASE
3575 0000130A 7536                      <1>         jne    short K26           ; INTERRUPT RETURN
3576                                    <1>         ;
3577                                    <1>         ;----- ALTERNATE SHIFT KEY RELEASED, GET THE VALUE INTO BUFFER
3578 0000130C A0[896A0000]              <1>         mov    al, [ALT_INPUT]
3579 00001311 B400                      <1>         mov    ah, 0               ; SCAN CODE OF 0
3580 00001313 8825[896A0000]            <1>         mov    [ALT_INPUT], ah     ; ZERO OUT THE FIELD
3581 00001319 3C00                      <1>         cmp    al, 0               ; WAS THE INPUT = 0?
3582 0000131B 7425                      <1>         je     short K26           ; INTERRUPT_RETURN
3583                                    <1>         ; 29/01/2016
```

```
3584                                    <1>       ;jmp   K61                ; IT WASN'T, SO PUT IN BUFFER
3585 0000131D E9AB020000                <1>       jmp    _K60
3586                                    <1>       ;
3587                                    <1> K24:                            ; BREAK-TOGGLE
3588 00001322 2025[866A0000]            <1>       and    [KB_FLAG_1], ah     ; INDICATE NO LONGER DEPRESSED
3589 00001328 EB18                      <1>       jmp    short K26           ; INTERRUPT_RETURN
3590                                    <1>       ;
3591                                    <1>       ;----- TEST FOR HOLD STATE
3592                                    <1>                                   ; AL, AH = SCAN CODE
3593                                    <1> K25:                              ; NO-SHIFT-FOUND
3594 0000132A 3C80                      <1>       cmp    al, 80h              ; TEST FOR BREAK KEY
3595 0000132C 7314                      <1>       jae    short K26            ; NOTHING FOR BREAK CHARS FROM HERE ON
3596 0000132E F605[866A0000]08          <1>       test   byte [KB_FLAG_1], HOLD_STATE ; ARE WE IN HOLD STATE
3597 00001335 7428                      <1>       jz     short K28            ; BRANCH AROUND TEST IF NOT
3598 00001337 3C45                      <1>       cmp    al, NUM_KEY
3599 00001339 7407                      <1>       je     short K26            ; CAN'T END HOLD ON NUM_LOCK
3600 0000133B 8025[866A0000]F7          <1>       and    byte [KB_FLAG_1], ~HOLD_STATE ; TURN OFF THE HOLD STATE BIT
3601                                    <1> K26:
3602 00001342 8025[886A0000]FC          <1>       and    byte [KB_FLAG_3], ~(LC_E0+LC_E1) ; RESET LAST CHAR H.C. FLAG
3603                                    <1> K26A:                             ; INTERRUPT-RETURN
3604 00001349 FA                        <1>       cli                         ; TURN OFF INTERRUPTS
3605 0000134A B020                      <1>       mov    al, EOI              ; END OF INTERRUPT COMMAND
3606 0000134C E620                      <1>       out    20h, al ;out INTA00, al ; SEND COMMAND TO INTERRUPT CONTROL PORT
3607                                    <1> K27:                              ; INTERRUPT-RETURN-NO-EOI
3608 0000134E B0AE                      <1>       mov    al, ENA_KBD          ; INSURE KEYBOARD IS ENABLED
3609 00001350 E8D3020000                <1>       call   SHIP_IT              ; EXECUTE ENABLE
3610                                    <1> K27A:
3611 00001355 FA                        <1>       cli                         ; DISABLE INTERRUPTS
3612                                    <1>       ;;mov  byte [intflg], 0 ; 07/01/2017 ;; 15/01/2017
3613 00001356 07                        <1>       pop    es                   ; RESTORE REGISTERS
3614 00001357 1F                        <1>       pop    ds
3615 00001358 5F                        <1>       pop    edi
3616 00001359 5E                        <1>       pop    esi
3617 0000135A 5A                        <1>       pop    edx
3618 0000135B 59                        <1>       pop    ecx
3619 0000135C 5B                        <1>       pop    ebx
3620 0000135D 58                        <1>       pop    eax
3621                                    <1>       ;pop   ebp
3622 0000135E CF                        <1>       iretd                       ; RETURN
3623                                    <1>
3624                                    <1>       ;----- NOT IN HOLD STATE
3625                                    <1> K28:                              ; NO-HOLD-STATE
3626 0000135F 3C58                      <1>       cmp    al, 88               ; TEST FOR OUT-OF-RANGE SCAN CODES
3627 00001361 77DF                      <1>       ja     short K26            ; IGNORE IF OUT-OF-RANGE
3628                                    <1>       ;
3629 00001363 F6C308                    <1>       test   bl, ALT_SHIFT        ; ARE WE IN ALTERNATE SHIFT
3630 00001366 740E                      <1>       jz     short K28A           ; IF NOT ALTERNATE
3631                                    <1>       ; 12/04/2021
3632                                    <1>       ;jz    K38
3633                                    <1>       ;
3634 00001368 F6C710                    <1>       test   bh, KBX              ; IS THIS THE ENCHANCED KEYBOARD?
3635 0000136B 740E                      <1>       jz     short K29            ; NO, ALT STATE IS REAL
3636                                    <1>        ;28/02/2015
3637 0000136D F605[866A0000]04          <1>       test   byte [KB_FLAG_1], SYS_SHIFT ; YES, IS SYSREQ KEY DOWN?
3638 00001374 7405                      <1>       jz     short K29            ;  NO, ALT STATE IS REAL
3639                                    <1>       ; 12/04/2021
3640                                    <1>       ;jnz   K38                  ; YES, THIS IS PHONY ALT STATE
3641                                    <1>         ;                         ; DUE TO PRESSING SYSREQ
3642 00001376 E9C4000000                <1> K28A: jmp    K38
3643                                    <1>       ;
3644                                    <1>       ;----- TEST FOR RESET KEY SEQUENCE (CTL ALT DEL)
3645                                    <1> K29:                              ; TEST-RESET
3646 0000137B F6C304                    <1>       test   bl, CTL_SHIFT        ; ARE WE IN CONTROL SHIFT ALSO?
3647 0000137E 740B                      <1>       jz     short K31            ; NO_RESET
3648 00001380 3C53                      <1>       cmp    al, DEL_KEY          ; CTL-ALT STATE, TEST FOR DELETE KEY
3649 00001382 7507                      <1>       jne    short K31            ; NO_RESET, IGNORE
3650                                    <1>       ;
3651                                    <1>       ;----- CTL-ALT-DEL HAS BEEN FOUND
3652                                    <1>       ; 26/08/2014
3653                                    <1> cpu_reset:
3654                                    <1>       ; IBM PC/AT ROM BIOS source code - 10/06/85 (TEST4.ASM - PROC_SHUTDOWN)
3655                                    <1>       ; Send FEh (system reset command) to the keyboard controller.
3656 00001384 B0FE                      <1>       mov    al, SHUT_CMD         ; SHUTDOWN COMMAND
3657 00001386 E664                      <1>       out    STATUS_PORT, al      ; SEND TO KEYBOARD CONTROL PORT
3658                                    <1> khere:
3659 00001388 F4                        <1>       hlt                         ; WAIT FOR 80286 RESET
3660 00001389 EBFD                      <1>       jmp    short khere          ; INSURE HALT
3661                                    <1>       ;
3662                                    <1>       ;----- IN ALTERNATE SHIFT, RESET NOT FOUND
3663                                    <1> K31:                              ; NO-RESET
3664 0000138B 3C39                      <1>       cmp    al, 57               ; TEST FOR SPACE KEY
3665 0000138D 7507                      <1>       jne    short K311           ; NOT THERE
3666 0000138F B020                      <1>       mov    al, ' '              ; SET SPACE CHAR
3667                                    <1> k31a: ; 12/04/2021
3668 00001391 E929020000                <1>       jmp    K57                  ; BUFFER_FILL
3669                                    <1> K311:
3670 00001396 3C0F                      <1>       cmp    al, 15               ; TEST FOR TAB KEY
3671 00001398 7506                      <1>       jne    short K312           ; NOT THERE
3672 0000139A 66B800A5                  <1>       mov    ax, 0A500h           ; SET SPECIAL CODE FOR ALT-TAB
3673                                    <1>       ;jmp   K57                  ; BUFFER_FILL
3674                                    <1>       ; 12/04/2021
3675 0000139E EBF1                      <1>       jmp    short k31a
3676                                    <1> K312:
3677 000013A0 3C4A                      <1>       cmp    al, 74               ; TEST FOR KEY PAD -
3678                                    <1>       ;je    short K37B           ; GO PROCESS
3679                                    <1>       ; 12/04/2021
3680 000013A2 7404                      <1>       je     short k312a
3681 000013A4 3C4E                      <1>       cmp    al, 78               ; TEST FOR KEY PAD +
3682                                    <1>       ;je    short K37B           ; GO PROCESS
3683                                    <1>       ; 12/04/2021
3684 000013A6 7505                      <1>       jne    short K32
3685                                    <1> k312a:
3686 000013A8 E988000000                <1>       jmp    K37B
3687                                    <1>       ;
3688                                    <1>       ;----- LOOK FOR KEY PAD ENTRY
```

```
                        <1> K32:                            ; ALT-KEY-PAD
000013AD BF[48690000]   <1>         mov    edi, K30         ; ALT-INPUT-TABLE offset
000013B2 B90A000000     <1>         mov    ecx, 10          ; LOOK FOR ENTRY USING KEYPAD
000013B7 F2AE           <1>         repne  scasb            ; LOOK FOR MATCH
000013B9 7521           <1>         jne    short K33        ; NO_ALT_KEYPAD
000013BB F6C702         <1>         test   bh, LC_E0        ; IS THIS ONE OF THE NEW KEYS?
000013BE 7579           <1>         jnz    short K37C       ; YES, JUMP, NOT NUMPAD KEY
000013C0 81EF[49690000] <1>         sub    edi, K30+1       ; DI NOW HAS ENTRY VALUE
000013C6 A0[896A0000]   <1>         mov    al, [ALT_INPUT]  ; GET THE CURRENT BYTE
000013CB B40A           <1>         mov    ah, 10           ; MULTIPLY BY 10
000013CD F6E4           <1>         mul    ah
000013CF 6601F8         <1>         add    ax, di           ; ADD IN THE LATEST ENTRY
000013D2 A2[896A0000]   <1>         mov    [ALT_INPUT], al  ; STORE IT AWAY
                        <1> K32A:
000013D7 E966FFFFFF     <1>         jmp   K26                 ; THROW AWAY THAT KEYSTROKE
                        <1>         ;
                        <1>         ;----- LOOK FOR SUPERSHIFT ENTRY
                        <1> K33:                            ; NO-ALT-KEYPAD
000013DC C605[896A0000]00 <1>       mov   byte [ALT_INPUT], 0 ; ZERO ANY PREVIOUS ENTRY INTO INPUT
000013E3 B91A000000     <1>         mov    ecx, 26          ; (DI),(ES) ALREADY POINTING
000013E8 F2AE           <1>         repne  scasb            ; LOOK FOR MATCH IN ALPHABET
000013EA 7445           <1>         je     short K37A       ; MATCH FOUND, GO FILLL THE BUFFER
                        <1>         ;
                        <1>         ;----- LOOK FOR TOP ROW OF ALTERNATE SHIFT
                        <1> K34:                            ; ALT-TOP-ROW
000013EC 3C02           <1>         cmp    al, 2            ; KEY WITH '1' ON IT
000013EE 7245           <1>         jb     short K37B       ; MUST BE ESCAPE
000013F0 3C0D           <1>         cmp    al, 13           ; IS IT IN THE REGION
000013F2 7705           <1>         ja     short K35        ; NO, ALT SOMETHING ELSE
000013F4 80C476         <1>         add    ah, 118          ; CONVERT PSEUDO SCAN CODE TO RANGE
000013F7 EB38           <1>         jmp    short K37A       ; GO FILL THE BUFFER
                        <1>         ;
                        <1>         ;----- TRANSLATE ALTERNATE SHIFT PSEUDO SCAN CODES
                        <1> K35:                            ; ALT-FUNCTION
000013F9 3C57           <1>         cmp    al, F11_M        ; IS IT F11?
000013FB 7209           <1>         jb     short K35A ; 20/02/2015   ; NO, BRANCH
000013FD 3C58           <1>         cmp    al, F12_M        ; IS IT F12?
000013FF 7705           <1>         ja     short K35A ; 20/02/2015   ; NO, BRANCH
00001401 80C434         <1>         add    ah, 52           ; CONVERT TO PSEUDO SCAN CODE
00001404 EB2B           <1>         jmp    short K37A       ; GO FILL THE BUFFER
                        <1> K35A:
00001406 F6C702         <1>         test   bh, LC_E0        ; DO WE HAVE ONE OF THE NEW KEYS?
00001409 741B           <1>         jz     short K37        ; NO, JUMP
0000140B 3C1C           <1>         cmp    al, 28           ; TEST FOR KEYPAD ENTER
0000140D 7506           <1>         jne   short K35B        ; NOT THERE
0000140F 66B800A6       <1>         mov    ax, 0A600h       ; SPECIAL CODE
                        <1>         ;jmp   K57              ; BUFFER FILL
                        <1>         ; 12/04/2021
00001413 EB0C           <1>         jmp    short k35c
                        <1> K35B:
00001415 3C53           <1>         cmp    al, 83           ; TEST FOR DELETE KEY
00001417 7420           <1>         je     short K37C       ; HANDLE WITH OTHER EDIT KEYS
00001419 3C35           <1>         cmp    al, 53           ; TEST FOR KEYPAD /
0000141B 75BA           <1>         jne    short K32A       ; NOT THERE, NO OTHER E0 SPECIALS
                        <1>         ; 12/04/2021
                        <1>         ;jne   K26
0000141D 66B800A4       <1>         mov    ax, 0A400h       ; SPECIAL CODE1
                        <1> k35c: ; 12/04/2021
00001421 E999010000     <1>         jmp    K57              ; BUFFER FILL
                        <1> K37:
00001426 3C3B           <1>         cmp    al, 59           ; TEST FOR FUNCTION KEYS (F1)
00001428 720B           <1>         jb    short K37B        ; NO FN, HANDLE W/OTHER EXTENDED
0000142A 3C44           <1>         cmp    al, 68           ; IN KEYPAD REGION?
0000142C 77A9           <1>         ja     short K32A       ; IF SO, IGNORE
                        <1>         ; 12/04/2021
                        <1>         ;ja    K26
0000142E 80C42D         <1>         add    ah, 45           ; CONVERT TO PSEUDO SCAN CODE
                        <1> K37A:
00001431 B000           <1>         mov    al, 0            ; ASCII CODE OF ZERO
                        <1>         ;jmp   K57              ; PUT IT IN THE BUFFER
                        <1>         ; 12/04/2021
00001433 EBEC           <1>         jmp    short k35c
                        <1> K37B:
00001435 B0F0           <1>         mov    al, 0F0h         ; USE SPECIAL ASCII CODE
                        <1>         ;jmp   K57              ; PUT IT IN THE BUFFER
                        <1>         ; 12/04/2021
00001437 EBE8           <1>         jmp    short k35c
                        <1> K37C:
00001439 0450           <1>         add    al, 80           ; CONVERT SCAN CODE (EDIT KEYS)
0000143B 88C4           <1>         mov    ah, al           ; (SCAN CODE NOT IN AH FOR INSERT)
0000143D EBF2           <1>         jmp     short K37A      ; PUT IT IN THE BUFFER
                        <1>         ;
                        <1>         ;----- NOT IN ALTERNATE SHIFT
                        <1> K38:                            ; NOT-ALT-SHIFT
                        <1>                                 ; BL STILL HAS SHIFT FLAGS
0000143F F6C304         <1>         test   bl, CTL_SHIFT     ; ARE WE IN CONTROL SHIFT?
                        <1>         ;;jnz  short K38A       ; YES, START PROCESSING
                        <1>         ;jz    K44              ; NOT-CTL-SHIFT
                        <1>         ; 12/04/2021
00001442 7505           <1>         jnz    short K38A       ; YES, START PROCESSING
00001444 E9AB000000     <1>         jmp    K44              ; NOT-CTL-SHIFT
                        <1>         ;
                        <1>         ;----- CONTROL SHIFT, TEST SPECIAL CHARACTERS
                        <1>         ;----- TEST FOR BREAK
                        <1> K38A:
00001449 3C46           <1>         cmp    al, SCROLL_KEY   ; TEST FOR BREAK
0000144B 7530           <1>         jne    short K39        ; JUMP, NO-BREAK
0000144D F6C710         <1>         test   bh, KBX          ; IS THIS THE ENHANCED KEYBOARD?
00001450 7405           <1>         jz     short K38B       ; NO, BREAK IS VALID
00001452 F6C702         <1>         test   bh, LC_E0        ; YES, WAS LAST CODE AN E0?
00001455 7426           <1>         jz     short K39        ; NO-BREAK, TEST FOR PAUSE
                        <1> K38B:
00001457 8B1D[926A0000] <1>         mov    ebx, [BUFFER_HEAD]  ; RESET BUFFER TO EMPTY
0000145D 891D[966A0000] <1>         mov    [BUFFER_TAIL], ebx
00001463 C605[846A0000]80 <1>       mov    byte [BIOS_BREAK], 80h  ; TURN ON BIOS_BREAK BIT
```

```
                                <1>       ;
                                <1>       ;----- ENABLE KEYBOARD
0000146A B0AE                   <1>       mov   al, ENA_KBD         ; ENABLE KEYBOARD
0000146C E8B7010000             <1>       call  SHIP_IT             ; EXECUTE ENABLE
                                <1>       ;
                                <1>       ; CTRL+BREAK code here !!!
                                <1>       ;INT  1BH                 ; BREAK INTERRUPT VECTOR
                                <1>       ; 17/10/2015
00001471 E85A5C0000             <1>       call  ctrlbrk ; control+break subroutine
                                <1>       ;
                                <1>       ;sub  ax, ax              ; PUT OUT DUMMY CHARACTER
                                <1>         ; 12/04/2021
00001476 29C0                   <1>       sub   eax, eax
00001478 E942010000             <1>       jmp   K57                 ; BUFFER_FILL
                                <1>       ;
                                <1>       ;----- TEST FOR PAUSE
                                <1> K39:                            ; NO_BREAK
0000147D F6C710                 <1>       test  bh, KBX                   ; IS THIS THE ENHANCED KEYBOARD?
00001480 7537                   <1>       jnz   short K41           ; YES, THEN THIS CAN'T BE PAUSE
00001482 3C45                   <1>       cmp   al, NUM_KEY         ; LOOK FOR PAUSE KEY
00001484 7533                   <1>       jne   short K41           ; NO-PAUSE
                                <1> K39P:
00001486 800D[866A0000]08       <1>       or    byte [KB_FLAG_1], HOLD_STATE ; TURN ON THE HOLD FLAG
                                <1>       ;
                                <1>       ;----- ENABLE KEYBOARD
0000148D B0AE                   <1>       mov   al, ENA_KBD         ; ENABLE KEYBOARD
0000148F E894010000             <1>       call  SHIP_IT             ; EXECUTE ENABLE
                                <1> K39A:
00001494 B020                   <1>       mov   al, EOI             ; END OF INTERRUPT TO CONTROL PORT
00001496 E620                   <1>       out   20h, al ;out INTA00, al ; ALLOW FURTHER KEYSTROKE INTERRUPTS
                                <1>       ;
                                <1>       ;----- DURING PAUSE INTERVAL, TURN COLOR CRT BACK ON
00001498 803D[BA6A0000]07       <1>       cmp    byte [CRT_MODE], 7 ; IS THIS BLACK AND WHITE CARD
0000149F 740A                   <1>       je    short K40           ; YES, NOTHING TO DO
000014A1 66BAD803               <1>       mov    dx, 03D8h          ; PORT FOR COLOR CARD
000014A5 A0[BB6A0000]           <1>       mov   al, [CRT_MODE_SET]  ; GET THE VALUE OF THE CURRENT MODE
000014AA EE                     <1>       out    dx, al             ; SET THE CRT MODE, SO THAT CRT IS ON
                                <1>       ;
                                <1> K40:                            ; PAUSE-LOOP
000014AB F605[866A0000]08       <1>       test  byte [KB_FLAG_1], HOLD_STATE ; CHECK HOLD STATE FLAG
000014B2 75F7                   <1>       jnz    short K40          ; LOOP UNTIL FLAG TURNED OFF
                                <1>       ;
000014B4 E995FEFFFF             <1>       jmp     K27               ; INTERRUPT_RETURN_NO_EOI
                                <1>         ;
                                <1>       ;----- TEST SPECIAL CASE KEY 55
                                <1> K41:                            ; NO-PAUSE
000014B9 3C37                   <1>       cmp    al, 55             ; TEST FOR */PRTSC KEY
000014BB 7513                   <1>       jne    short K42          ; NOT-KEY-55
000014BD F6C710                 <1>       test   bh, KBX            ; IS THIS THE ENHANCED KEYBOARD?
000014C0 7405                   <1>       jz     short K41A         ; NO, CTL-PRTSC IS VALID
000014C2 F6C702                 <1>       test   bh, LC_E0          ; YES, WAS LAST CODE AN E0?
000014C5 7421                   <1>       jz     short K42B         ; NO, TRANSLATE TO A FUNCTION
                                <1> K41A:
000014C7 66B80072               <1>       mov    ax, 114*256        ; START/STOP PRINTING SWITCH
000014CB E9EF000000             <1>       jmp   K57                 ; BUFFER_FILL
                                <1>       ;
                                <1>       ;----- SET UP TO TRANSLATE CONTROL SHIFT
                                <1> K42:                            ; NOT-KEY-55
000014D0 3C0F                   <1>       cmp    al, 15             ; IS IT THE TAB KEY?
000014D2 7414                   <1>       je     short K42B         ; YES, XLATE TO FUNCTION CODE
000014D4 3C35                   <1>       cmp    al, 53             ; IS IT THE / KEY?
000014D6 750E                   <1>       jne    short K42A         ; NO, NO MORE SPECIAL CASES
000014D8 F6C702                 <1>       test   bh, LC_E0          ; YES, IS IT FROM THE KEY PAD?
000014DB 7409                   <1>       jz     short K42A         ; NO, JUST TRANSLATE
000014DD 66B80095               <1>       mov    ax, 9500h          ; YES, SPECIAL CODE FOR THIS ONE
000014E1 E9D9000000             <1>       jmp    K57                ; BUFFER FILL
                                <1> K42A:
                                <1>       ;;mov  ebx, _K8           ; SET UP TO TRANSLATE CTL
000014E6 3C3B                   <1>       cmp    al, 59             ; IS IT IN CHARACTER TABLE?
                                <1>       ;jb    short K45F          ; YES, GO TRANSLATE CHAR
                                <1>       ;;jb   K56 ; 20/02/2015
                                <1>       ;;jmp  K64 ; 20/02/2015
                                <1> K42B:
000014E8 BB[7C690000]           <1>       mov    ebx, _K8           ; SET UP TO TRANSLATE CTL
                                <1>       ;jb    K56 ;; 20/02/2015
                                <1>       ; 12/04/2021
000014ED 7267                   <1>       jb     short K45F
000014EF E9B9000000             <1>       jmp    K64
                                <1>         ;
                                <1>       ;----- NOT IN CONTROL SHIFT
                                <1> K44:                            ; NOT-CTL-SHIFT
000014F4 3C37                   <1>       cmp    al, 55             ; PRINT SCREEN KEY?
000014F6 7528                   <1>       jne    short K45          ; NOT PRINT SCREEN
000014F8 F6C710                 <1>       test   bh, KBX            ; IS THIS THE ENHANCED KEYBOARD?
000014FB 7407                   <1>       jz     short K44A         ; NO, TEST FOR SHIFT STATE
000014FD F6C702                 <1>       test   bh, LC_E0          ; YES, LAST CODE A MARKER?
00001500 7507                   <1>       jnz    short K44B         ; YES, IS PRINT SCREEN
00001502 EB41                   <1>       jmp    short K45C         ; NO, TRANSLATE TO '*' CHARACTER
                                <1> K44A:
00001504 F6C303                 <1>       test   bl, LEFT_SHIFT+RIGHT_SHIFT ; NOT 101 KBD, SHIFT KEY DOWN?
00001507 743C                   <1>       jz     short K45C         ; NO, TRANSLATE TO '*' CHARACTER
                                <1>       ;
                                <1>       ;----- ISSUE INTERRUPT TO INDICATE PRINT SCREEN FUNCTION
                                <1> K44B:
00001509 B0AE                   <1>       mov    al, ENA_KBD        ; INSURE KEYBOARD IS ENABLED
0000150B E818010000             <1>       call   SHIP_IT            ; EXECUTE ENABLE
00001510 B020                   <1>       mov    al, EOI            ; END OF CURRENT INTERRUPT
00001512 E620                   <1>       out    20h, al ;out INTA00, al   ; SO FURTHER THINGS CAN HAPPEN
                                <1>       ; Print Screen !!!        ; ISSUE PRINT SCREEN INTERRUPT (INT 05h)
                                <1>       ;PUSH  BP                 ; SAVE POINTER
                                <1>       ;INT   5H                 ; ISSUE PRINT SCREEN INTERRUPT
                                <1>       ;POP   BP                 ; RESTORE POINTER
00001514 8025[886A0000]FC       <1>       and   byte [KB_FLAG_3], ~(LC_E0+LC_E1) ; ZERO OUT THESE FLAGS
0000151B E92EFEFFFF             <1>       jmp   K27                 ; GO BACK WITHOUT EOI OCCURRING
                                <1>       ;
```

```
3899                            <1>       ;----- HANDLE IN-CORE KEYS
3900                            <1> K45:                          ; NOT-PRINT-SCREEN
3901 00001520 3C3A              <1>       cmp    al, 58           ; TEST FOR IN-CORE AREA
3902 00001522 7734              <1>       ja     short K46        ; JUMP IF NOT
3903 00001524 3C35              <1>       cmp    al, 53           ; IS THIS THE '/' KEY?
3904 00001526 7505              <1>       jne    short K45A       ; NO, JUMP
3905 00001528 F6C702            <1>       test   bh, LC_E0        ; WAS THE LAST CODE THE MARKER?
3906 0000152B 7518              <1>       jnz    short K45C       ; YES, TRANSLATE TO CHARACTER
3907                            <1> K45A:
3908 0000152D B91A000000        <1>       mov    ecx, 26          ; LENGHT OF SEARCH
3909 00001532 BF[52690000]      <1>       mov    edi, K30+10      ; POINT TO TABLE OF A-Z CHARS
3910 00001537 F2AE              <1>       repne  scasb            ; IS THIS A LETTER KEY?
3911                            <1>       ; 20/02/2015
3912 00001539 7505              <1>       jne    short K45B       ; NO, SYMBOL KEY
3913                            <1>       ;
3914 0000153B F6C340            <1>       test   bl, CAPS_STATE   ; ARE WE IN CAPS_LOCK?
3915 0000153E 750C              <1>       jnz    short K45D       ; TEST FOR SURE
3916                            <1> K45B:
3917 00001540 F6C303            <1>       test   bl, LEFT_SHIFT+RIGHT_SHIFT ; ARE WE IN SHIFT STATE?
3918 00001543 750C              <1>       jnz    short K45E       ; YES, UPPERCASE
3919                            <1>                               ; NO, LOWERCASE
3920                            <1> K45C:
3921 00001545 BB[D4690000]      <1>       mov    ebx, K10         ; TRANSLATE TO LOWERCASE LETTERS
3922 0000154A EB51              <1>       jmp    short K56
3923                            <1> K45D:                         ; ALMOST-CAPS-STATE
3924 0000154C F6C303            <1>       test   bl, LEFT_SHIFT+RIGHT_SHIFT ; CL ON. IS SHIFT ON, TOO?
3925 0000154F 75F4              <1>       jnz    short K45C       ; SHIFTED TEMP OUT OF CAPS STATE
3926                            <1> K45E:
3927 00001551 BB[2C6A0000]      <1>       mov    ebx, K11         ; TRANSLATE TO UPPER CASE LETTERS
3928 00001556 EB45              <1> K45F: jmp    short K56
3929                            <1>       ;
3930                            <1>       ;----- TEST FOR KEYS F1 - F10
3931                            <1> K46:                          ; NOT IN-CORE AREA
3932 00001558 3C44              <1>       cmp    al, 68           ; TEST FOR F1 - F10
3933                            <1>       ;ja    short K47        ; JUMP IF NOT
3934                            <1>       ;jmp   short K53        ; YES, GO DO FN KEY PROCESS
3935 0000155A 7635              <1>       jna    short K53
3936                            <1>       ;
3937                            <1>       ;----- HANDLE THE NUMERIC PAD KEYS
3938                            <1> K47:                          ; NOT F1 - F10
3939 0000155C 3C53              <1>       cmp    al, 83           ; TEST NUMPAD KEYS
3940 0000155E 772D              <1>       ja     short K52        ; JUMP IF NOT
3941                            <1>       ;
3942                            <1>       ;----- KEYPAD KEYS, MUST TEST NUM LOCK FOR DETERMINATION
3943                            <1> K48:
3944 00001560 3C4A              <1>       cmp    al, 74           ; SPECIAL CASE FOR MINUS
3945 00001562 74ED              <1>       je     short K45E       ; GO TRANSLATE
3946 00001564 3C4E              <1>       cmp    al, 78           ; SPECIAL CASE FOR PLUS
3947 00001566 74E9              <1>       je     short K45E       ; GO TRANSLATE
3948 00001568 F6C702            <1>       test   bh, LC_E0        ; IS THIS ONE OFTHE NEW KEYS?
3949 0000156B 750A              <1>       jnz    short K49        ; YES, TRANSLATE TO BASE STATE
3950                            <1>       ;
3951 0000156D F6C320            <1>       test   bl, NUM_STATE    ; ARE WE IN NUM LOCK
3952 00001570 7514              <1>       jnz    short K50        ; TEST FOR SURE
3953 00001572 F6C303            <1>       test   bl, LEFT_SHIFT+RIGHT_SHIFT ; ARE WE IN SHIFT STATE?
3954                            <1>       ;jnz   short K51        ; IF SHIFTED, REALLY NUM STATE
3955 00001575 75DA              <1>       jnz    short K45E
3956                            <1>       ;
3957                            <1>       ;----- BASE CASE FOR KEYPAD
3958                            <1> K49:
3959 00001577 3C4C              <1>       cmp    al, 76           ; SPECIAL CASE FOR BASE STATE 5
3960 00001579 7504              <1>       jne    short K49A       ; CONTINUE IF NOT KEYPAD 5
3961 0000157B B0F0              <1>       mov    al, 0F0h         ; SPECIAL ASCII CODE
3962 0000157D EB40              <1>       jmp    short K57        ; BUFFER FILL
3963                            <1> K49A:
3964 0000157F BB[D4690000]      <1>       mov    ebx, K10         ; BASE CASE TABLE
3965 00001584 EB27              <1>       jmp    short K64        ; CONVERT TO PSEUDO SCAN
3966                            <1>       ;
3967                            <1>       ;----- MIGHT BE NUM LOCK, TEST SHIFT STATUS
3968                            <1> K50:                          ; ALMOST-NUM-STATE
3969 00001586 F6C303            <1>       test  bl, LEFT_SHIFT+RIGHT_SHIFT
3970 00001589 75EC              <1>       jnz    short K49        ; SHIFTED TEMP OUT OF NUM STATE
3971 0000158B EBC4              <1> K51:  jmp    short K45E       ; REALLY NUM STATE
3972                            <1>       ;
3973                            <1>       ;----- TEST FOR THE NEW KEYS ON WT KEYBOARDS
3974                            <1> K52:                          ; NOT A NUMPAD KEY
3975 0000158D 3C56              <1>       cmp    al, 86           ; IS IT THE NEW WT KEY?
3976                            <1>       ;jne   short K53        ; JUMP IF NOT
3977                            <1>       ;jmp   short K45B       ; HANDLE WITH REST OF LETTER KEYS
3978 0000158F 74AF              <1>       je     short K45B
3979                            <1>       ;
3980                            <1>       ;----- MUST BE F11 OR F12
3981                            <1> K53:                          ; F1 - F10 COME HERE, TOO
3982 00001591 F6C303            <1>       test   bl, LEFT_SHIFT+RIGHT_SHIFT ; TEST SHIFT STATE
3983 00001594 74E1              <1>       jz     short K49        ; JUMP, LOWER CASE PSEUDO SC'S
3984                            <1>       ; 20/02/2015
3985 00001596 BB[2C6A0000]      <1>       mov    ebx, K11         ; UPPER CASE PSEUDO SCAN CODES
3986 0000159B EB10              <1>       jmp    short K64        ; TRANSLATE SCAN
3987                            <1>       ;
3988                            <1>       ;----- TRANSLATE THE CHARACTER
3989                            <1> K56:                          ; TRANSLATE-CHAR
3990 0000159D FEC8              <1>       dec    al               ; CONVERT ORIGIN
3991 0000159F D7                <1>       xlat                    ; CONVERT THE SCAN CODE TO ASCII
3992 000015A0 F605[886A0000]02  <1>       test   byte [KB_FLAG_3], LC_E0 ; IS THIS A NEW KEY?
3993 000015A7 7416              <1>       jz     short K57        ; NO, GO FILL BUFFER
3994 000015A9 B4E0              <1>       mov    ah, MC_E0        ; YES, PUT SPECIAL MARKER IN AH
3995 000015AB EB12              <1>       jmp    short K57        ; PUT IT INTO THE BUFFER
3996                            <1>       ;
3997                            <1>       ;----- TRANSLATE SCAN FOR PSEUDO SCAN CODES
3998                            <1> K64:                          ; TRANSLATE-SCAN-ORGD
3999 000015AD FEC8              <1>       dec    al               ; CONVERT ORIGIN
4000 000015AF D7                <1>       xlat                    ; CTL TABLE SCAN
4001 000015B0 88C4              <1>       mov    ah, al           ; PUT VALUE INTO AH
4002 000015B2 B000              <1>       mov    al, 0            ; ZERO ASCII CODE
4003 000015B4 F605[886A0000]02  <1>       test   byte [KB_FLAG_3], LC_E0   ; IS THIS A NEW KEY?
```

```
4004 000015BB 7402                <1>       jz     short K57           ; NO, GO FILL BUFFER
4005 000015BD B0E0                <1>       mov    al, MC_E0           ; YES, PUT SPECIAL MARKER IN AL
4006                              <1>       ;
4007                              <1>       ;----- PUT CHARACTER INTO BUFFER
4008                              <1> K57:                             ; BUFFER_FILL
4009 000015BF 3CFF                <1>       cmp    al, -1              ; IS THIS AN IGNORE CHAR
4010 000015C1 7405                <1>       je     short K59           ; YES, DO NOTHING WITH IT
4011                              <1>       ;je    K26                 ; YES, DO NOTHING WITH IT
4012 000015C3 80FCFF              <1>       cmp    ah, -1              ; LOOK FOR -1 PSEUDO SCAN
4013                              <1>       ;;jne  short K61           ; NEAR_INTERRUPT_RETURN
4014                              <1>       ;je    K26                 ; INTERRUPT_RETURN
4015                              <1>       ; 12/04/2021
4016 000015C6 7505                <1>       jne    short _K60          ; NEAR_INTERRUPT_RETURN
4017                              <1> K59:                             ; NEAR_INTERRUPT_RETURN
4018 000015C8 E975FDFFFF          <1>       jmp    K26                 ; INTERRUPT_RETURN
4019                              <1>
4020                              <1> _K60: ; 29/01/2016
4021 000015CD 80FC68              <1>       cmp    ah, 68h      ; ALT + F1 key
4022 000015D0 721F                <1>       jb     short K61
4023 000015D2 80FC6F              <1>       cmp    ah, 6Fh ; ALT + F8 key
4024 000015D5 771A                <1>       ja     short K61
4025                              <1>       ;
4026 000015D7 8A1D[D6800100]      <1>       mov    bl, [ACTIVE_PAGE]
4027 000015DD 80C368              <1>       add    bl, 68h
4028 000015E0 38E3                <1>       cmp    bl, ah
4029 000015E2 740D                <1>       je     short K61
4030 000015E4 6650                <1>       push   ax
4031 000015E6 88E0                <1>       mov    al, ah
4032 000015E8 2C68                <1>       sub    al, 68h
4033 000015EA E83C090000          <1>       call   set_active_page
4034 000015EF 6658                <1>       pop    ax
4035                              <1> K61:                             ; NOT-CAPS-STATE
4036 000015F1 8B1D[966A0000]      <1>       mov    ebx, [BUFFER_TAIL]  ; GET THE END POINTER TO THE BUFFER
4037 000015F7 89DE                <1>       mov    esi, ebx            ; SAVE THE VALUE
4038 000015F9 E8AFFAFFFF          <1>       call   _K4                 ; ADVANCE THE TAIL
4039 000015FE 3B1D[926A0000]      <1>       cmp    ebx, [BUFFER_HEAD]  ; HAS THE BUFFER WRAPPED AROUND
4040 00001604 740E                <1>       je     short K62           ; BUFFER_FULL_BEEP
4041 00001606 668906              <1>       mov    [esi], ax           ; STORE THE VALUE
4042 00001609 891D[966A0000]      <1>       mov    [BUFFER_TAIL], ebx  ; MOVE THE POINTER UP
4043 0000160F E92EFDFFFF          <1>       jmp    K26
4044                              <1>       ;;cli                      ; TURN OFF INTERRUPTS
4045                              <1>       ;;mov  al, EOI             ; END OF INTERRUPT COMMAND
4046                              <1>       ;;out  INTA00, al          ; SEND COMMAND TO INTERRUPT CONTROL PORT
4047                              <1>       ;MOV   AL, ENA_KBD         ; INSURE KEYBOARD IS ENABLED
4048                              <1>       ;CALL  SHIP_IT             ; EXECUTE ENABLE
4049                              <1>       ;MOV   AX, 9102H           ; MOVE IN POST CODE & TYPE
4050                              <1>       ;INT   15H                 ; PERFORM OTHER FUNCTION
4051                              <1>       ;;and  byte [KB_FLAG_3],~(LC_E0+LC_E1) ; RESET LAST CHAR H.C. FLAG
4052                              <1>       ;JMP   K27A                ; INTERRUPT_RETURN
4053                              <1>       ;;jmp   K27
4054                              <1>       ;
4055                              <1>       ;----- BUFFER IS FULL SOUND THE BEEPER
4056                              <1> K62:
4057 00001614 B020                <1>       mov    al, EOI             ; ENABLE INTERRUPT CONTROLLER CHIP
4058 00001616 E620                <1>       out    INTA00, al
4059 00001618 66B9A602            <1>       mov    cx, 678             ; DIVISOR FOR 1760 HZ
4060 0000161C B304                <1>       mov    bl, 4               ; SHORT BEEP COUNT (1/16 + 1/64 DELAY)
4061 0000161E E8380D0000          <1>       call   beep                ; GO TO COMMON BEEP HANDLER
4062 00001623 E926FDFFFF          <1>       jmp   K27                  ; EXIT
4063                              <1>
4064                              <1> SHIP_IT:
4065                              <1>       ;--------------------------------------------------------------------------------
4066                              <1>       ; SHIP_IT
4067                              <1>       ;      THIS ROUTINES HANDLES TRANSMISSION OF COMMAND AND DATA BYTES
4068                              <1>       ;      TO THE KEYBOARD CONTROLLER.
4069                              <1>       ;--------------------------------------------------------------------------------
4070                              <1>       ;
4071                              <1>
4072                              <1>       ;push  ax                  ; SAVE DATA TO SEND
4073                              <1>       ; 12/04/2021
4074 00001628 50                  <1>       push   eax
4075                              <1>
4076                              <1>       ;----- WAIT FOR COMMAND TO ACCEPTED
4077 00001629 FA                  <1>       cli                        ; DISABLE INTERRUPTS TILL DATA SENT
4078                              <1>       ; xor  ecx, ecx            ; CLEAR TIMEOUT COUNTER
4079 0000162A B900000100          <1>       mov    ecx, 10000h
4080                              <1> S10:
4081 0000162F E464                <1>       in     al, STATUS_PORT     ; READ KEYBOARD CONTROLLER STATUS
4082 00001631 A802                <1>       test   al, INPT_BUF_FULL   ; CHECK FOR ITS INPUT BUFFER BUSY
4083 00001633 E0FA                <1>       loopnz S10                 ; WAIT FOR COMMAND TO BE ACCEPTED
4084                              <1>
4085                              <1>       ;pop   ax                  ; GET DATA TO SEND
4086                              <1>       ; 12/04/2021
4087 00001635 58                  <1>       pop    eax
4088                              <1>
4089 00001636 E664                <1>       out    STATUS_PORT, al     ; SEND TO KEYBOARD CONTROLLER
4090 00001638 FB                  <1>       sti                        ; ENABLE INTERRUPTS AGAIN
4091 00001639 C3                  <1>       retn                       ; RETURN TO CALLER
4092                              <1>
4093                              <1>       ; 12/04/2021 (32 bit push/pop)
4094                              <1> SND_DATA:
4095                              <1>       ; --------------------------------------------------------------------------------
4096                              <1>       ; SND_DATA
4097                              <1>       ;      THIS ROUTINES HANDLES TRANSMISSION OF COMMAND AND DATA BYTES
4098                              <1>       ;      TO THE KEYBOARD AND RECEIPT OF ACKNOWLEDGEMENTS. IT ALSO
4099                              <1>       ;      HANDLES ANY RETRIES IF REQUIRED
4100                              <1>       ; --------------------------------------------------------------------------------
4101                              <1>       ;
4102 0000163A 50                  <1>       push   eax ; push ax       ; SAVE REGISTERS
4103 0000163B 53                  <1>       push   ebx ; push bx
4104 0000163C 51                  <1>       push   ecx
4105 0000163D 88C7                <1>       mov    bh, al              ; SAVE TRANSMITTED BYTE FOR RETRIES
4106 0000163F B303                <1>       mov    bl, 3               ; LOAD RETRY COUNT
4107                              <1> SD0:
4108 00001641 FA                  <1>       cli                        ; DISABLE INTERRUPTS
```

```
4109 00001642 8025[876A0000]CF    <1>       and   byte [KB_FLAG_2], ~(KB_FE+KB_FA) ; CLEAR ACK AND RESEND FLAGS
4110                              <1>       ;
4111                              <1>       ;----- WAIT FOR COMMAND TO BE ACCEPTED
4112 00001649 B900000100          <1>       mov   ecx, 10000h         ; MAXIMUM WAIT COUNT
4113                              <1> SD5:
4114 0000164E E464                <1>       in    al, STATUS_PORT     ; READ KEYBOARD PROCESSOR STATUS PORT
4115 00001650 A802                <1>       test  al, INPT_BUF_FULL   ; CHECK FOR ANY PENDING COMMAND
4116 00001652 E0FA                <1>       loopnz SD5                ; WAIT FOR COMMAND TO BE ACCEPTED
4117                              <1>       ;
4118 00001654 88F8                <1>       mov   al, bh              ; REESTABLISH BYTE TO TRANSMIT
4119 00001656 E660                <1>       out   PORT_A, al          ; SEND BYTE
4120 00001658 FB                  <1>       sti                       ; ENABLE INTERRUPTS
4121                              <1>       ;mov  cx, 01A00h          ; LOAD COUNT FOR 10 ms+
4122 00001659 B9FFFF0000          <1>       mov   ecx, 0FFFFh
4123                              <1> SD1:
4124 0000165E F605[876A0000]30    <1>       test  byte [KB_FLAG_2], KB_FE+KB_FA ; SEE IF EITHER BIT SET
4125 00001665 750F                <1>       jnz   short SD3           ; IF SET, SOMETHING RECEIVED GO PROCESS
4126 00001667 E2F5                <1>       loop  SD1                 ; OTHERWISE WAIT
4127                              <1> SD2:
4128 00001669 FECB                <1>       dec   bl                  ; DECREMENT RETRY COUNT
4129 0000166B 75D4                <1>       jnz   short SD0           ; RETRY TRANSMISSION
4130 0000166D 800D[876A0000]80    <1>       or    byte [KB_FLAG_2], KB_ERR ; TURN ON TRANSMIT ERROR FLAG
4131 00001674 EB09                <1>       jmp   short SD4           ; RETRIES EXHAUSTED FORGET TRANSMISSION
4132                              <1> SD3:
4133 00001676 F605[876A0000]10    <1>       test  byte [KB_FLAG_2], KB_FA ; SEE IF THIS IS AN ACKNOWLEDGE
4134 0000167D 74EA                <1>       jz    short SD2           ; IF NOT, GO RESEND
4135                              <1> SD4:
4136 0000167F 59                  <1>       pop   ecx                 ; RESTORE REGISTERS
4137 00001680 5B                  <1>       pop   ebx ; pop bx
4138 00001681 58                  <1>       pop   eax ; pop ax
4139 00001682 C3                  <1>       retn                      ; RETURN, GOOD TRANSMISSION
4140                              <1>
4141                              <1> SND_LED:
4142                              <1>       ; ------------------------------------------------------------------------------------
4143                              <1>       ; SND_LED
4144                              <1>       ;     THIS ROUTINES TURNS ON THE MODE INDICATORS.
4145                              <1>       ;
4146                              <1>       ;------------------------------------------------------------------------------------
4147                              <1>       ;
4148 00001683 FA                  <1>       cli                       ; TURN OFF INTERRUPTS
4149 00001684 F605[876A0000]40    <1>       test  byte [KB_FLAG_2], KB_PR_LED ; CHECK FOR MODE INDICATOR UPDATE
4150 0000168B 755F                <1>       jnz   short SL1           ; DON'T UPDATE AGAIN IF UPDATE UNDERWAY
4151                              <1>       ;
4152 0000168D 800D[876A0000]40    <1>       or    byte [KB_FLAG_2], KB_PR_LED ; TURN ON UPDATE IN PROCESS
4153 00001694 B020                <1>       mov   al, EOI             ; END OF INTERRUPT COMMAND
4154 00001696 E620                <1>       out   20h, al ;out INTA00, al ; SEND COMMAND TO INTERRUPT CONTROL PORT
4155 00001698 EB11                <1>       jmp   short SL0           ; GO SEND MODE INDICATOR COMMAND
4156                              <1> SND_LED1:
4157 0000169A FA                  <1>       cli                       ; TURN OFF INTERRUPTS
4158 0000169B F605[876A0000]40    <1>       test  byte [KB_FLAG_2], KB_PR_LED ; CHECK FOR MODE INDICATOR UPDATE
4159 000016A2 7548                <1>       jnz   short SL1           ; DON'T UPDATE AGAIN IF UPDATE UNDERWAY
4160                              <1>       ;
4161 000016A4 800D[876A0000]40    <1>       or    byte [KB_FLAG_2], KB_PR_LED ; TURN ON UPDATE IN PROCESS
4162                              <1> SL0:
4163 000016AB B0ED                <1>       mov   al, LED_CMD         ; LED CMD BYTE
4164 000016AD E888FFFFFF          <1>       call  SND_DATA            ; SEND DATA TO KEYBOARD
4165 000016B2 FA                  <1>       cli
4166 000016B3 E836000000          <1>       call  MAKE_LED            ; GO FORM INDICATOR DATA BYTE
4167 000016B8 8025[876A0000]F8    <1>       and   byte [KB_FLAG_2], 0F8h ; ~KB_LEDS ; CLEAR MODE INDICATOR BITS
4168 000016BF 0805[876A0000]      <1>       or    [KB_FLAG_2], al     ; SAVE PRESENT INDICATORS FOR NEXT TIME
4169 000016C5 F605[876A0000]80    <1>       test  byte [KB_FLAG_2], KB_ERR ; TRANSMIT ERROR DETECTED
4170 000016CC 750F                <1>       jnz   short SL2           ; IF SO, BYPASS SECOND BYTE TRANSMISSION
4171                              <1>       ;
4172 000016CE E867FFFFFF          <1>       call  SND_DATA            ; SEND DATA TO KEYBOARD
4173 000016D3 FA                  <1>       cli                       ; TURN OFF INTERRUPTS
4174 000016D4 F605[876A0000]80    <1>       test  byte [KB_FLAG_2], KB_ERR ; TRANSMIT ERROR DETECTED
4175 000016DB 7408                <1>       jz    short SL3           ; IF NOT, DON'T SEND AN ENABLE COMMAND
4176                              <1> SL2:
4177 000016DD B0F4                <1>       mov   al, KB_ENABLE       ; GET KEYBOARD CSA ENABLE COMMAND
4178 000016DF E856FFFFFF          <1>       call  SND_DATA            ; SEND DATA TO KEYBOARD
4179 000016E4 FA                  <1>       cli                       ; TURN OFF INTERRUPTS
4180                              <1> SL3:
4181 000016E5 8025[876A0000]3F    <1>       and   byte [KB_FLAG_2], ~(KB_PR_LED+KB_ERR) ; TURN OFF MODE INDICATOR
4182                              <1> SL1:                            ; UPDATE AND TRANSMIT ERROR FLAG
4183 000016EC FB                  <1>       sti                       ; ENABLE INTERRUPTS
4184 000016ED C3                  <1>       retn                      ; RETURN TO CALLER
4185                              <1>
4186                              <1> MAKE_LED:
4187                              <1>       ;------------------------------------------------------------------------------------
4188                              <1>       ; MAKE_LED
4189                              <1>       ;     THIS ROUTINES FORMS THE DATA BYTE NECESSARY TO TURN ON/OFF
4190                              <1>       ;     THE MODE INDICATORS.
4191                              <1>       ;------------------------------------------------------------------------------------
4192                              <1>       ;
4193                              <1>       ;push  cx                 ; SAVE CX
4194 000016EE A0[856A0000]        <1>       mov   al, [KB_FLAG]       ; GET CAPS & NUM LOCK INDICATORS
4195 000016F3 2470                <1>       and   al, CAPS_STATE+NUM_STATE+SCROLL_STATE ; ISOLATE INDICATORS
4196                              <1>       ;mov  cl, 4               ; SHIFT COUNT
4197                              <1>       ;rol  al, cl              ; SHIFT BITS OVER TO TURN ON INDICATORS
4198 000016F5 C0C004              <1>       rol   al, 4 ; 20/02/2015
4199 000016F8 2407                <1>       and   al, 07h             ; MAKE SURE ONLY MODE BITS ON
4200                              <1>       ;pop   cx
4201 000016FA C3                  <1>       retn                      ; RETURN TO CALLER
4202                              <1>
4203                              <1> ; % include 'kybdata.s'   ; KEYBOARD DATA
4204                              <1>
4205                              <1>
4206                              <1> ; /// End Of KEYBOARD FUNCTIONS ///
2678
2679                                  %include 'video.s' ; 07/03/2015
2680                              <1> ; ****************************************************************************
2681                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.4 - video.s
2682                              <1> ; ----------------------------------------------------------------------------
2683                              <1> ; Last Update: 17/04/2021
2684                              <1> ; ----------------------------------------------------------------------------
```

```
                        <1> ; Beginning: 16/01/2016
                        <1> ; ------------------------------------------------------------------------------
                        <1> ; Assembler: NASM version 2.15 (trdos386.s)
                        <1> ; ------------------------------------------------------------------------------
                        <1> ; Turkish Rational DOS
                        <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
                        <1> ;
                        <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
                        <1> ; video.inc (13/08/2015)
                        <1> ;
                        <1> ; Derived from 'IBM PC-AT' BIOS source code (1985)
                        <1> ; ****************************************************************************
                        <1>
                        <1> ; Retro UNIX 386 v1 Kernel - VIDEO.INC
                        <1> ; Last Modification: 13/08/2015
                        <1> ;             (Video Data is in 'VIDATA.INC')
                        <1> ;
                        <1> ; ///////// VIDEO (CGA) FUNCTIONS ////////////////
                        <1>
                        <1> ; 16/01/2016 (32 bit modifications, TRDOS386 - TRDOS v2.0, video.s)
                        <1> ; INT 31H (TRDOS 386) = INT 10H (IBM PC/AT REAL MODE)
                        <1>
                        <1> ; IBM PC-AT BIOS Source Code
                        <1> ; TITLE VIDEO1 --- 06/10/85 VIDEO DISPLAY BIOS
                        <1>
                        <1> _int10h:
                        <1>       ; 23/03/2016
                        <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
000016FB 9C             <1>       pushfd
000016FC 0E             <1>       push   cs
000016FD E851000000     <1>       call   VIDEO_IO_1
00001702 C3             <1>       retn
                        <1>
                        <1> ;--- INT 10 H ----------------------------------------------------------------
                        <1> ; VIDEO_IO                                                                  :
                        <1> ;     THESE ROUTINES PROVIDE THE CRT DISPLAY INTERFACE                      :
                        <1> ;     THE FOLLOWING FUNCTIONS ARE PROVIDED:                                 :
                        <1> ;                                                                           :
                        <1> ;    (AH)= 00H     SET MODE (AL) CONTAINS MODE VALUE                        :
                        <1> ;           (AL) = 00H  40X25 BW MODE (POWER ON DEFAULT)                    :
                        <1> ;           (AL) = 01H  40X25 COLOR                                         :
                        <1> ;           (AL) = 02H  80X25 BW                                            :
                        <1> ;           (AL) = 03H  80X25 COLOR                                         :
                        <1> ;                         GRAPHICS MODES                                    :
                        <1> ;           (AL) = 04H  320X200 COLOR                                       :
                        <1> ;           (AL) = 05H  320X200 BW MODE                                     :
                        <1> ;           (AL) = 06H  640X200 BW MODE                                     :
                        <1> ;           (AL) = 07H   80X25 MONOCHROME (USED INTERNAL TO VIDEO ONLY)     :
                        <1> ;          *** NOTES -BW MODES OPERATE SAME AS COLOR MODES, BUT COLOR        :
                        <1> ;                      BURST IS NOT ENABLED                                 :
                        <1> ;                     -CURSOR IS NOT DISPLAYED IN GRAPHICS MODE             :
                        <1> ;    (AH)= 01H     SET CURSOR TYPE                                          :
                        <1> ;           (CH) = BITS 4-0 = START LINE FOR CURSOR                         :
                        <1> ;                  ** HARDWARE WILL ALWAYS CAUSE BLINK                      :
                        <1> ;                  ** SETTING BIT 5 OR 6 WILL CAUSE ERRATIC BLINKING        :
                        <1> ;                     OR NO CURSOR AT ALL                                   :
                        <1> ;           (CL) = BITS 4-0 = END LINE FOR CURSOR                           :
                        <1> ;    (AH)= 02H     SET CURSOR POSITION                                      :
                        <1> ;           (DH,DL) = ROW,COLUMN  (00H,00H) IS UPPER LEFT                   :
                        <1> ;           (BH) = A PAGE NUMBER (MUST BE 00H FOR GRAPHICS MODES)           :
                        <1> ;    (AH)= 03H     READ CURSOR POSITION                                     :
                        <1> ;           (BH) = PAGE NUMBER (MUST BE 00H FOR GRAPHICS MODES)             :
                        <1> ;          ON EXIT (DH,DL) = ROW,COLUMN OF CURRENT CURSOR                   :
                        <1> ;                  (CH,CL) = CURSOR MODE CURRENTLY SET                      :
                        <1> ;    (AH)= 04H     READ LIGHT PEN POSITION                                  :
                        <1> ;          ON EXIT:                                                         :
                        <1> ;           (AH) = 00H -- LIGHT PEN SWITCH NOT DOWN/NOT TRIGGERED           :
                        <1> ;           (AH) = 01H -- VALID LIGHT PEN VALUE IN REGISTERS                :
                        <1> ;                  (DH,DL) = ROW,COLUMN OF CHARACTER LP POSITION            :
                        <1> ;                  (CH) = RASTER LINE (0-199)                               :
                        <1> ;                  (BX) = PIXEL COLUMN (0-319,639)                          :
                        <1> ;    (AH)= 05H     SELECT ACTIVE DISPLAY PAGE (VALID ONLY FOR ALPHA MODES):
                        <1> ;           (AL) = NEW PAGE VALUE (0-7 FOR MODES 0&1, 0-3 FOR MODES 2&3):
                        <1> ;    (AH)= 06H     SCROLL ACTIVE PAGE UP                                    :
                        <1> ;           (AL) = NUMBER OF LINES. ( LINES BLANKED AT BOTTOM OF WINDOW ):
                        <1> ;                  (AL) = 00H MEANS BLANK ENTIRE WINDOW                     :
                        <1> ;           (CH,CL) = ROW,COLUMN OF UPPER LEFT CORNER OF SCROLL             :
                        <1> ;           (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL            :
                        <1> ;           (BH) = ATTRIBUTE TO BE USED ON BLANK LINE                       :
                        <1> ;    (AH)= 07H     SCROLL ACTIVE PAGE DOWN                                  :
                        <1> ;           (AL) = NUMBER OF LINES, INPUT LINES BLANKED AT TOP OF WINDOW:
                        <1> ;                  (AL) = 00H MEANS BLANK ENTIRE WINDOW                     :
                        <1> ;           (CH,CL) = ROW,COLUMN OF UPPER LEFT CORNER OF SCROLL             :
                        <1> ;           (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL            :
                        <1> ;           (BH) = ATTRIBUTE TO BE USED ON BLANK LINE                       :
                        <1> ;                                                                           :
                        <1> ;   CHARACTER HANDLING ROUTINES                                             :
                        <1> ;                                                                           :
                        <1> ;    (AH)= 08H     READ ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION      :
                        <1> ;           (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)                :
                        <1> ;          ON EXIT:                                                         :
                        <1> ;           (AL) = CHAR READ                                                :
                        <1> ;           (AH) = ATTRIBUTE OF CHARACTER READ (ALPHA MODES ONLY)            :
                        <1> ;    (AH)= 09H     WRITE ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION:
                        <1> ;           (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)                 :
                        <1> ;           (CX) = COUNT OF CHARACTERS TO WRITE                             :
                        <1> ;           (AL) = CHAR TO WRITE                                            :
                        <1> ;           (BL) = ATTRIBUTE OF CHARACTER (ALPHA)/COLOR OF CHAR (GRAPHICS):
                        <1> ;                    SEE NOTE ON WRITE DOT FOR BIT 7 OF BL = 1.             :
                        <1> ;    (AH) = 0AH    WRITE CHARACTER ONLY AT CURRENT CURSOR POSITION           :
                        <1> ;           (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)                 :
                        <1> ;           (CX) = COUNT OF CHARACTERS TO WRITE                             :
                        <1> ;           (AL) = CHAR TO WRITE                                            :
                        <1> ;                  NOTE: USE FUNCTION (AH)= 09H IN GRAPHICS MODES           :
                        <1> ;    FOR READ/WRITE CHARACTER INTERFACE WHILE IN GRAPHICS MODE, THE          :
```

```
                                <1> ;           CHARACTERS ARE FORMED FROM A CHARACTER GENERATOR IMAGE     :
                                <1> ;           MAINTAINED IN THE SYSTEM ROM. ONLY THE 1ST 128 CHARS       :
                                <1> ;           ARE CONTAINED THERE. TO READ/WRITE THE SECOND 128 CHARS,    :
                                <1> ;           THE USER MUST INITIALIZE THE POINTER AT INTERRUPT 1FH       :
                                <1> ;           (LOCATION 0007CH) TO POINT TO THE 1K BYTE TABLE CONTAINING :
                                <1> ;           THE CODE POINTS FOR THE SECOND 128 CHARS (128-255).        :
                                <1> ;     FOR WRITE CHARACTER INTERFACE IN GRAPHICS MODE, THE REPLICATION FACTOR:
                                <1> ;           CONTAINED IN (CX) ON ENTRY WILL PRODUCE VALID RESULTS ONLY :
                                <1> ;           FOR CHARACTERS CONTAINED ON THE SAME ROW. CONTINUATION TO   :
                                <1> ;           SUCCEEDING LINES WILL NOT PRODUCE CORRECTLY.                :
                                <1> ;                                                                       :
                                <1> ;    GRAPHICS INTERFACE                                                 :
                                <1> ;    (AH)= 0BH     SET COLOR PALETTE                                    :
                                <1> ;           (BH) = PALETTE COLOR ID BEING SET (0-127)                   :
                                <1> ;           (BL) = COLOR VALUE TO BE USED WITH THAT COLOR ID            :
                                <1> ;                  NOTE: FOR THE CURRENT COLOR CARD, THIS ENTRY POINT HAS:
                                <1> ;                          MEANING ONLY FOR 320X200 GRAPHICS.          :
                                <1> ;                  COLOR ID = 0 SELECTS THE BACKGROUND COLOR (0-15)     :
                                <1> ;                  COLOR ID = 1 SELECTS THE PALETTE TO BE USED:         :
                                <1> ;                          0 = GREEN(1)/RED(2)/YELLOW(3)               :
                                <1> ;                          1 = CYAN(1)/MAGENTA(2)/WHITE(3)             :
                                <1> ;                  IN 40X25 OR 80X25 ALPHA MODES, THE VALUE SET FOR     :
                                <1> ;                          PALETTE COLOR 0 INDICATES THE BORDER COLOR   :
                                <1> ;                          TO BE USED (VALUES 0-31, WHERE 16-31 SELECT  :
                                <1> ;                          THE HIGH INTENSITY BACKGROUND SET.           :
                                <1> ;    (AH)= 0CH     WRITE DOT                                            :
                                <1> ;           (DX) = ROW NUMBER                                           :
                                <1> ;           (CX) = COLUMN NUMBER                                        :
                                <1> ;           (AL) = COLOR VALUE                                          :
                                <1> ;                   IF BIT 7 OF AL = 1, THEN THE COLOR VALUE IS EXCLUSIVE:
                                <1> ;                   ORed WITH THE CURRENT CONTENTS OF THE DOT            :
                                <1> ;    (AH)= ODH     READ DOT                                             :
                                <1> ;           (DX) = ROW NUMBER                                           :
                                <1> ;           (CX) = COLUMN NUMBER                                        :
                                <1> ;           (AL) = RETURNS THE DOT READ                                 :
                                <1> ;                                                                       :
                                <1> ;    ASCII TELETYPE ROUTINE FOR OUTPUT                                  :
                                <1> ;                                                                       :
                                <1> ;    (AH)= 0EH     WRITE TELETYPE TO ACTIVE PAGE                        :
                                <1> ;           (AL) = CHAR TO WRITE                                        :
                                <1> ;           (BL) = FOREGROUND COLOR IN GRAPHICS MODE                    :
                                <1> ;           NOTE -- SCREEN WIDTH IS CONTROLLED BY PREVIOUS MODE SET     :
                                <1> ;    (AH)= 0FH     CURRENT VIDEO STATE                                  :
                                <1> ;           RETURNS THE CURRENT VIDEO STATE                             :
                                <1> ;           (AL) = MODE CURRENTLY SET ( SEE (AH)=00H FOR EXPLANATION)   :
                                <1> ;           (AH) = NUMBER OR CHARACTER COLUMNS ON SCREEN                 :
                                <1> ;           (BH) = CURRENT ACTIVE DISPLAY PAGE                          :
                                <1> ;    (AH)= 10H     RESERVED                                             :
                                <1> ;    (AH)= 11H     RESERVED                                             :
                                <1> ;    (AH)= 12H     RESERVED                                             :
                                <1> ;    (AH)= 13H     WRITE STRING                                         :
                                <1> ;                  ES:BP  -  POINTER T0 STRING TO BE WRITTEN            :
                                <1> ;                  CX     -  LENGTH OF CHARACTER STRING TO WRITTEN      :
                                <1> ;                  DX     -  CURSOR POSITION FOR STRING TO BE WRITTEN   :
                                <1> ;                  BH     -  PAGE NUMBER                               :
                                <1> ;           (AL)= 00H    WRITE CHARACTER STRING                         :
                                <1> ;                  BL     -  ATTRIBUTE                                 :
                                <1> ;                  STRING IS  <CHAR,CHAR, ... ,CHAR>                   :
                                <1> ;                  CURSOR NOT MOVED                                    :
                                <1> ;           (AL)= 01H    WRITE CHARACTER STRING AND MOVE CURSOR         :
                                <1> ;                  BL     -  ATTRIBUTE                                 :
                                <1> ;                  STRING IS  <CHAR,CHAR, ... ,CHAR>                   :
                                <1> ;                  CURSOR MOVED                                        :
                                <1> ;           (AL)= 02H    WRITE CHARACTER AND ATTRIBUTE STRING           :
                                <1> ;                         (VALID FOR ALPHA MODES ONLY)                  :
                                <1> ;                  STRING IS <CHAR,ATTR,CHAR,ATTR ..  ,CHAR,ATTR>       :
                                <1> ;                  CURSOR IS NOT MOVED                                  :
                                <1> ;           (AL)= 03H WRITE CHARACTER AND ATTRIBUTE STRING AND MOVE CURSOR:
                                <1> ;                         (VALID FOR ALPHA MODES ONLY)                  :
                                <1> ;                  STRING IS <CHAR,ATTR,CHAR,ATTR ..  ,CHAR,ATTR>       :
                                <1> ;                  CURSOR IS MOVED                                      :
                                <1> ;            NOTE:  CARRIAGE RETURN, LINE FEED, BACKSPACE, AND BELL ARE:
                                <1> ;                          TREATED AS COMMANDS RATHER THAN PRINTABLE CHARACTERS.:
                                <1> ;                                                                       :
                                <1> ;     BX,CX,DX,SI,DI,BP,SP,DS,ES,SS PRESERVED DURING CALLS EXCEPT FOR   :
                                <1> ;     BX,CX,DX RETURN VALUES ON FUNCTIONS 03H,04H,0DH AND 0FH. ON ALL CALLS:
                                <1> ;     AX IS MODIFIED.                                                   :
                                <1> ;-------------------------------------------------------------------------
                                <1>
00001703 [B41A0000]             <1> M1:   dd    SET_MODE     ; TABLE OF ROUTINES WITHIN VIDEO I/O
00001707 [6B1E0000]             <1>       dd    SET_CTYPE
0000170B [9F1E0000]             <1>       dd    SET_CPOS
0000170F [C71E0000]             <1>       dd    READ_CURSOR
                                <1>       ;dd   VIDEO_RETURN ; READ_LPEN
00001713 [B21A0000]             <1>       dd    set_mode_ncm ; Set mode without clearing video memory
00001717 [0D1F0000]             <1>       dd    ACT_DISP_PAGE
0000171B [9A1F0000]             <1>       dd    SCROLL_UP
0000171F [BA200000]             <1>       dd    SCROLL_DOWN
00001723 [39210000]             <1>       dd    READ_AC_CURRENT
00001727 [96210000]             <1>       dd    WRITE_AC_CURRENT
0000172B [BC210000]             <1>       dd    WRITE_C_CURRENT
0000172F [EB2A0000]             <1>       dd    SET_COLOR
00001733 [562B0000]             <1>       dd    WRITE_DOT
00001737 [212B0000]             <1>       dd    READ_DOT
0000173B [46220000]             <1>       dd    WRITE_TTY
0000173F [9A1A0000]             <1>       dd    VIDEO_STATE
00001743 [DE350000]             <1>       dd    vga_pal_funcs; 10/08/2016 (TRDOS 386)
00001747 [57300000]             <1>       dd    font_setup   ; 10/07/2016 (TRDOS 386)
0000174B [E91A0000]             <1>       dd    VIDEO_RETURN ; RESERVED
0000174F [BB230000]             <1>       dd    WRITE_STRING ; 23/06/2016 (TRDOS 386)
                                <1> M1L   EQU   $ - M1
                                <1>
                                <1> ; 06/12/2020
                                <1> ; 05/12/2020
                                <1> ; 03/12/2020
```

```
                        <1> ; 27/11/2020 - TRDOS 386 v2.0.3
                        <1> ; 14/01/2017
                        <1> ; 02/01/2017
                        <1> ; 04/07/2016
                        <1> ; 12/05/2016
                        <1> ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                        <1> int31h:  ; Video BIOS
                        <1>
                        <1> ; BH = Video page number
                        <1> ; BL = Color/Attribute
                        <1> ; AH = Function number
                        <1> ; AL = Character
                        <1>
                        <1> VIDEO_IO_1:
                        <1>       ;sti                      ; INTERRUPTS BACK ON
00001753 FC             <1>       cld                       ; SET DIRECTION FORWARD
                        <1>
                        <1>       ;cmp   ah, M1L/4          ; TEST FOR WITHIN TABLE RANGE
                        <1>       ;jnb   short M4           ; BRANCH TO EXIT IF NOT A VALID COMMAND
                        <1>
                        <1>       ; 26/11/2020
00001754 80FC14         <1>       cmp    ah, M1L/4
00001757 7205           <1>       jb     short VGA_func
                        <1>
00001759 80FC4F         <1>       cmp    ah, 4Fh
0000175C 7532           <1>       jne    short M4 ; invalid !
                        <1>
                        <1> VGA_func: ; 26/11/2020
0000175E 06             <1>       push   es ; *
0000175F 1E             <1>       push   ds ; **            ; SAVE WORK AND PARAMETER REGISTERS
                        <1>
                        <1>       ; 26/11/2020
00001760 50             <1>       push   eax ; -
                        <1>
00001761 66B81000       <1>       mov    ax, KDATA          ; POINT DS: TO DATA SEGMENT
00001765 8ED8           <1>       mov    ds, ax
00001767 8EC0           <1>       mov    es, ax
                        <1>
                        <1>       ; 26/11/2020
00001769 58             <1>       pop    eax ; +
                        <1>       ;
0000176A FB             <1>       sti
0000176B 80FC4F         <1>       cmp    ah, 4Fh
0000176E 747A           <1>       je     short VBE_func
                        <1>
                        <1>       ; 04/12/2020
00001770 A3[308D0100]   <1>       mov    [video_eax], eax
                        <1>
                        <1>       ; 21/12/2020
00001775 803D[BA6A0000]FF <1>     cmp    byte [CRT_MODE], 0FFh ; Current mode is a VESA VBE mode ?
0000177C 7213           <1>       jb     short VGA_func_std
                        <1>
0000177E 08E4           <1>       or     ah, ah             ; set mode ?
00001780 750F           <1>       jnz    short VGA_func_std ; no
                        <1>
00001782 803D[4E090000]03 <1>     cmp    byte [vbe3], 3     ; (real) VESA VBE 3 video bios ?
00001789 7506           <1>       jne    short VGA_func_std ; no
                        <1>
0000178B E989010000     <1>       jmp    vesa_vbe3_pmi
                        <1>
                        <1>       ; 21/12/2020
                        <1> M4:                              ; COMMAND NOT VALID
00001790 CF             <1>       iretd                      ; DO NOTHING IF NOT IN VALID RANGE
                        <1>
                        <1> VGA_func_std:
                        <1>       ; 06/12/2020
                        <1>       ; 03/12/2020
00001791 80FC0F         <1>       cmp    ah, 0Fh
00001794 773D           <1>       ja     short VGA_funcs_0   ; only CGA funcs will be handled by
                        <1>       ; 06/12/2020               ; vga bios firmware
                        <1>       ; 03/12/2020
                        <1>       ;test  ah, 7Fh ; set mode ?
                        <1>       ;;or   ah, ah              ; only 'set mode' will be handled by
                        <1>       ;jnz   short VGA_funcs_0   ; vga bios firmware
                        <1>       ;jz    short vbe_pmi32_0
                        <1>
                        <1>       ; 28/11/2020
00001796 803D[04120300]00 <1>     cmp    byte [pmi32], 0     ; 32 bit protected mode interface for
0000179D 7634           <1>       jna    short VGA_funcs_0   ; video hardware's vga bios firmware
                        <1>                                  ; ([pmi32] > 0 if it is activated)
                        <1>                                  ; note:
                        <1>                                  ; [vbe3] = 3 is required to activate
                        <1>       ; 07/12/2020
0000179F 20E4           <1>       and    ah, ah ; is this set mode ?
000017A1 7413           <1>       jz     short vbe_pmi32_2 ; yes
                        <1>
000017A3 80FC04         <1>       cmp    ah, 04h       ; set mode ('no clear memory' option)
000017A6 742B           <1>       je     short VGA_funcs_0
                        <1>
                        <1>       ; 07/12/2020
000017A8 803D[BA6A0000]07 <1>     cmp    byte [CRT_MODE], 7 ; current mode > 7 ?
000017AF 7622           <1>       jna    short VGA_funcs_0  ; no
                        <1>
                        <1>       ; when mode 3 is active,
                        <1>       ; video bios functions are not redirected
                        <1>       ; to VESA VBE3 PMI except 'set mode' function
                        <1>
                        <1> vbe_pmi32_0:
                        <1>       ; 06/12/2020
                        <1>       ;or    ah, ah
                        <1>       ;jnz   short vbe_pmi32_2
                        <1>       ; ah = 0 ; 'set mode'
                        <1>       ;cmp   al, 3 ; 80x25 text mode, 16 colors, default mode for MainProg
                        <1>       ;jne   short vbe_pmi32_1
                        <1>       ;cmp   byte [CRT_MODE], 3 ; If current video mode <> 3 and requested
```

```
3000                                    <1>     ;                   ; video mode is 3, use internal 'set mode';
3001                                    <1>     ;                   ; otherwise, use vesa vbe 3 bios's 'set mode'.
3002                                    <1>     ;jne   short VGA_funcs_0
3003                                    <1> vbe_pmi32_1:
3004 000017B1 E963010000                <1>     jmp    vesa_vbe3_pmi
3005                                    <1> ;vbe_pmi32_2:
3006                                    <1>     ;cmp   ah, 04h ; set mode (no clear mem option)
3007                                    <1>     ;jne   short vbe_pmi32_1
3008                                    <1>
3009                                    <1> vbe_pmi32_2:
3010                                    <1>     ; 07/12/2020
3011 000017B6 803D[BA6A0000]07          <1>     cmp    byte [CRT_MODE], 7 ; current mode > 7 ?
3012 000017BD 77F2                      <1>     ja     short vbe_pmi32_1  ; yes
3013                                    <1>
3014 000017BF 3C07                      <1>     cmp    al, 7  ; requested mode > 7 ?
3015 000017C1 7610                      <1>     jna    short VGA_funcs_0  ; no (CGA)
3016                                    <1>
3017 000017C3 3C13                      <1>     cmp    al, 13h
3018 000017C5 76EA                      <1>     jna    short vbe_pmi32_1
3019                                    <1>
3020 000017C7 A880                      <1>     test   al, 80h
3021 000017C9 7408                      <1>     jz     short VGA_funcs_0  ; unknown or special
3022                                    <1>
3023 000017CB 3C87                      <1>     cmp    al, 87h      ; requested mode > 7 ?
3024                                    <1>                         ; (with no clear mem ops)
3025 000017CD 7604                      <1>     jna    short VGA_funcs_0  ; no (CGA)
3026                                    <1>
3027 000017CF 3C93                      <1>     cmp    al, 93h
3028 000017D1 76DE                      <1>     jna    short vbe_pmi32_1
3029                                    <1>
3030                                    <1>     ; > 13h video modes are unknown or special
3031                                    <1>     ; they must be handled by kernel
3032                                    <1>
3033                                    <1>     ; CGA video modes will be handled by kernel
3034                                    <1>
3035                                    <1> VGA_funcs_0:
3036 000017D3 52                        <1>     push   edx ; ***
3037 000017D4 51                        <1>     push   ecx ; ****
3038 000017D5 53                        <1>     push   ebx ; *****
3039 000017D6 56                        <1>     push   esi ; ******
3040 000017D7 57                        <1>     push   edi ; *******
3041 000017D8 55                        <1>     push   ebp ; ********
3042                                    <1>
3043                                    <1>     ;mov   [video_eax], eax ; 12/05/2016
3044 000017D9 BF00800B00                <1>     mov    edi, 0B8000h        ; GET offset FOR COLOR CARD
3045                                    <1>
3046                                    <1>     ; 23/03/2016
3047 000017DE C0E402                    <1>     shl    ah, 2  ; dword            ; TIMES 2 FOR WORD TABLE LOOKUP
3048 000017E1 0FB6F4                    <1>     movzx  esi, ah                   ; MOVE OFFSET INTO LOOK UP REGISTER (SI)
3049                                    <1>     ;mov   ah, [CRT_MODE]            ; MOVE CURRENT MODE INTO (AH) REGISTER
3050                                    <1>
3051                                    <1>     ;;15/01/2017
3052                                    <1>     ; 14/01/2017
3053                                    <1>     ; 02/01/2017
3054                                    <1>     ;;mov  byte [intflg], 31h  ; video interrupt
3055                                    <1>     ;sti ; 26/11/2020
3056                                    <1>     ;
3057                                    <1>
3058 000017E4 FFA6[03170000]            <1>     JMP    dword [esi+M1]            ; GO TO SELECTED FUNCTION
3059                                    <1>
3060                                    <1> VBE_func:
3061                                    <1>     ; 26/11/2020
3062                                    <1>     ;sti
3063 000017EA 55                        <1>     push   ebp ; *** ; 27/11/2020
3064 000017EB 56                        <1>     push   esi ; ****
3065                                    <1>
3066                                    <1>     ; Note:
3067                                    <1>     ; ebx, ecx, edx, edi, ebp registers
3068                                    <1>     ; must be saved by VBE functions and
3069                                    <1>     ; eax register must be set
3070                                    <1>     ; (according to VBE 3 standard)
3071                                    <1>     ; before return from this interrupt
3072                                    <1>     ; (every function must restore and set
3073                                    <1>     ; registers except esp, esi, es, ds)
3074                                    <1>
3075 000017EC 803D[4E090000]02          <1>     cmp    byte [vbe3], 2
3076 000017F3 7741                      <1>     ja     short VESA_VBE3_PMI_CALL ; VBE3 video bios ('PMID')
3077                                    <1>     ;je    short VBE_func_0  ; Bochs/Qemu/VirtualBox emulator
3078 000017F5 726A                      <1>     jb     short VBE_unknown ; invalid ([vbe3] = 0)
3079                                    <1>
3080                                    <1>     ;jmp   VESA_VBE3_PMI_CALL
3081                                    <1>
3082                                    <1> VBE_func_0:
3083                                    <1>     ; Bochs/Plex86 VGAbios VBE extension
3084                                    <1>     ; (TRDOS 386 v2.0.3 can use VBE graphics modes on emulators)
3085                                    <1>     ; BOCHS/QEMU/VIRTUALBOX
3086                                    <1>
3087 000017F7 8A25[4F090000]            <1>     mov    ah, [vbe2bios]
3088 000017FD 80FCC0                    <1>     cmp    ah, 0C0h
3089 00001800 725F                      <1>     jb     short VBE_unknown
3090 00001802 80FCC5                    <1>     cmp    ah, 0C5h
3091 00001805 775A                      <1>     ja     short VBE_unknown
3092                                    <1>
3093                                    <1>     ; TRDOS 386 is running on BOCHS or QEMU
3094 00001807 B44F                      <1>     mov    ah, 4Fh
3095                                    <1> VBE_func_1:
3096 00001809 0FB6F0                    <1>     movzx  esi, al ; VESA VBE function number
3097 0000180C 66C1E602                  <1>     shl    si, 2 ; dword
3098 00001810 6683FE14                  <1>     cmp    si, N1L
3099 00001814 734B                      <1>     jnb    short VBE_unknown
3100                                    <1>     ;sti
3101                                    <1>
3102 00001816 FF96[22180000]            <1>     call   dword [esi+N1] ; call VBE function
3103                                    <1>
3104                                    <1>     ;jmp   short VBE_bios_return
```

```
                                <1>
                                <1> VBE_bios_return:
0000181C FA                     <1>         cli
0000181D 5E                     <1>         pop    esi ; ****
0000181E 5D                     <1>         pop    ebp ; *** ; 27/11/2020
0000181F 07                     <1>         pop    es  ; **
00001820 1F                     <1>         pop    ds  ; *
00001821 CF                     <1>         iretd
                                <1>
                                <1>         ; 26/11/2020
                                <1> N1:
00001822 [85180000]             <1>         dd     vbe_biosfn_return_ctrl_info
00001826 [62390000]             <1>         dd     vbe_biosfn_return_mode_info
0000182A [1D3A0000]             <1>         dd     vbe_biosfn_set_mode
0000182E [ED3A0000]             <1>         dd     vbe_biosfn_return_current_mode
00001832 [0C3B0000]             <1>         dd     vbe_biosfn_save_restore_state
                                <1>         ;dd    vbe_biosfn_display_window_ctrl
                                <1>         ;dd    vbe_biosfn_set_get_log_scanline
                                <1>         ;dd    vbe_biosfn_set_get_disp_start
                                <1>         ;dd    vbe_biosfn_set_get_dac_pal_frm
                                <1>         ;dd    vbe_biosfn_set_get_palette_data
                                <1>
                                <1> N1L   EQU    $ - N1
                                <1>
                                <1>
                                <1> VESA_VBE3_PMI_CALL: ; VESA VBE video bios (firmware) functions
                                <1>                     ; by using VESA VBE3 Protected Mode Interface
                                <1>
                                <1>         ; 29/11/2020
                                <1>         ; 26/11/2020 - TRDOS 386 v2.0.3 video.s
                                <1>
                                <1>         ; We are here because..
                                <1>         ; 'PMID' has been verified by TRDOS 386 v2.0.3 kernel.
                                <1>         ; (Otherwise bochs/plex86 compatible VBE functions and
                                <1>         ;  modes would be used on BOCHS/QEMU/VIRTUALBOX emulators
                                <1>         ;  or only standard/old VGA graphics modes would be used.)
                                <1>
                                <1>         ; (TRDOS 386 v2.0.3 can use VESA VBE graphics modes if
                                <1>         ;  the video bios is full compatible with VBE3 standard)
                                <1>
                                <1>         ; 29/11/2020
                                <1>
00001836 0FB6F0                 <1>         movzx  esi, al ; VESA VBE 3 function number
00001839 66C1E602               <1>         shl    si, 2 ; dword
0000183D 6683FE14               <1>         cmp    si, P1L
00001841 731E                   <1>         jnb    short VBE_unknown
                                <1>         ;sti
                                <1>
00001843 57                     <1>         push   edi ; *****
                                <1>
00001844 FF96[4D180000]         <1>         call   dword [esi+P1] ; call VBE 3 function
                                <1>
0000184A 5F                     <1>         pop    edi ; *****
                                <1>
0000184B EBCF                   <1>         jmp    short VBE_bios_return
                                <1>
                                <1> P1:
0000184D [67180000]             <1>         dd     vbe3_pmfn_return_ctrl_info
00001851 [8B180000]             <1>         dd     vbe3_pmfn_return_mode_info
00001855 [C8180000]             <1>         dd     vbe3_pmfn_set_mode
00001859 [C3180000]             <1>         dd     vbe3_pmfn_return_current_mode
0000185D [BA190000]             <1>         dd     vbe3_pmfn_save_restore_state
                                <1>         ;dd    vbe3_pmfn_display_window_ctrl
                                <1>         ;dd    vbe3_pmfn_set_get_log_scanline
                                <1>         ;dd    vbe3_pmfn_set_get_disp_start
                                <1>         ;dd    vbe3_pmfn_set_get_dac_pal_frm
                                <1>         ;dd    vbe3_pmfn_set_get_palette_data
                                <1>         ;dd    vbe3_pmfn_return_pmi ; invalid for TRDOS 386 v2
                                <1>         ;dd    vbe3_pmfn_set_get_pixel_clock
                                <1>
                                <1> P1L   EQU    $ - P1
                                <1>
                                <1> ;     ; 29/11/2020
                                <1> ;     mov    edi, VBE3MODEINFOBLOCK >> 4 ; / 16
                                <1> ;
                                <1> ;     cmp    al, 04h
                                <1> ;     jb     short vbe3_pm_f ; function: 4F00h to 4F03h
                                <1> ;     ja     short vbe3_pmi_f5B ; function: 4F05h to 4F0Bh
                                <1> ;
                                <1> ;     ; check buffer length (must be <= 2048 bytes)
                                <1> ;
                                <1> ;     and    dl, dl ; 0
                                <1> ;     jz     short vbe3_pm_f
                                <1> ;                   ; Return Save/Restore State buffer size
                                <1> ;
                                <1> ;     push   ebx  ; buffer address
                                <1> ;     push   edx  ; function: save (01h) or restore (02h)
                                <1> ;     call
                                <1> ;
                                <1> ;vbe3_pm_f03:
                                <1> ;     cmp    al, 2
                                <1> ;     ja     short vbe3_pm_f ; function 4F03h
                                <1> ;     jb     short vbe3_pm_f1
                                <1> ;
                                <1> ;
                                <1> ;vbe3_pm_f1:
                                <1> ;
                                <1> ;
                                <1> ;vbe3_pmi_f5B:
                                <1> ;     cmp    al, 09h
                                <1> ;     jna    short vbe3_pm_f ; funcs 05h to 09h are usable
                                <1> ;
                                <1> ;     cmp    al, 0Bh ; Get/Set pixel clock, last function
                                <1> ;     jne    short VBE_unknown
                                <1> ;                   ; (do not use 'uncertain' functions
```

```
                        <1> ;                   ; because of system-user buff transfers)
                        <1> ;vbe3_pm_f:
                        <1> ;       mov     byte [vbe3_pm_fn], al       ; set
                        <1> ;       ; prepare 16 bit pm segments & registers for pmi call
                        <1> ;       call    VESA_VBE3_PM_FUNCTION
                        <1> ;
                        <1> ;
                        <1>
                        <1>         ; 26/11/2020
                        <1> VBE_unknown:
00001861 66B80001       <1>         mov     ax, 0100h  ; ah = 1 : Function call failed
                        <1>                            ; al = 0 : Function is not supported
                        <1>
00001865 EBB5           <1>         jmp     short VBE_bios_return
                        <1>
                        <1> vbe3_pmfn_return_ctrl_info:
                        <1>         ; 12/12/2020
                        <1>         ;
                        <1>         ; VBE function 4F00h - Return VBE Controller Information
                        <1>         ;
                        <1>         ; Input:
                        <1>         ;       EDI = Pointer to buffer in which to place
                        <1>         ;             VbeInfoBlock structure
                        <1>         ;
                        <1>         ;       AX = 4F00h
                        <1>         ; Output:
                        <1>         ;       AX = VBE return status
                        <1>         ;            AX = 004Fh -> succeeded
                        <1>         ;            AX <> 004Fh -> failed
                        <1>         ;
                        <1>         ; Modified registers: eax (+ edi for kernel's own call)
                        <1>
                        <1>         ; NOTE: TRDOS 386 v2 (v2.0.3) kernel calls this function
                        <1>         ; during startup while cpu is in real mode
                        <1>         ; (by using int 10h, 4F02h) and saves VbeInfoBlock at
                        <1>         ; VBE3INFOBLOCK address (97E00h for TRDOS 386 v2.0.3).
                        <1>         ;
                        <1>         ; So...
                        <1>         ; This VBE function is adjusted to return/move same info
                        <1>         ; from VBE3INFOBLOCK to user's buffer in EDI.
                        <1>
                        <1>         ; int 31h (int 10h) entrance
                        <1>
00001867 21FF           <1>         and     edi, edi
00001869 7424           <1>         jz      short vbe3_func_fail ; invalid buffer address !
                        <1>
                        <1> ;_vbe3_pmfn_return_ctrl_info:
                        <1>         ;or     edi, edi
                        <1>         ;jnz    short _vbe_biosfn_return_ctrl_info
                        <1>         ;
                        <1>         ;; this option may not be necessary - 12/12/2020
                        <1>         ;
                        <1>         ;; edi = 0, kernel forces to get ctrl info again
                        <1>         ;;        by using VESA VBE3 video bios's pmi
                        <1>         ;
                        <1>         ;;push edi
                        <1>         ;; far call to VESA VBE3 PMI
                        <1>         ;;mov  ax, 4F00h ; Return VBE Controller Info
                        <1>         ;mov   edi, VBE3INFOBLOCK-VBE3SAVERESTOREBLOCK
                        <1>         ;; ES selector base address = VBE3SAVERESTOREBLOCK
                        <1>         ;call  int10h_32bit_pmi
                        <1>         ;;pop  edi
                        <1>         ;mov   edi, VBE3INFOBLOCK ; retn to the kernel sub
                        <1>         ;cmp   ax, 004Fh
                        <1>         ;je    short vbe_ctrl_info_retn
                        <1>         ;stc
                        <1>         ;retn
                        <1>
                        <1> _vbe_biosfn_return_ctrl_info:
0000186B 57             <1>         push    edi
0000186C 51             <1>         push    ecx
0000186D BE007E0900     <1>         mov     esi, VBE3INFOBLOCK
00001872 B900020000     <1>         mov     ecx, 512
00001877 E8C6FC0000     <1>         call    transfer_to_user_buffer
0000187C 59             <1>         pop     ecx
0000187D 5F             <1>         pop     edi
0000187E 720F           <1>         jc      short vbe3_func_fail
                        <1>
00001880 31C0           <1>         xor     eax, eax
00001882 B04F           <1>         mov     al, 4Fh ; successful
                        <1> ;vbe_ctrl_info_retn:
00001884 C3             <1>         retn
                        <1>
                        <1> vbe_biosfn_return_ctrl_info:
                        <1>         ; 12/12/2020
                        <1>         ;
                        <1>         ; VBE function 4F00h - Return VBE Controller Information
                        <1>         ;
                        <1>         ; Input:
                        <1>         ;       EDI = Pointer to buffer in which to place
                        <1>         ;             VbeInfoBlock structure
                        <1>         ;
                        <1>         ;       AX = 4F00h
                        <1>         ; Output:
                        <1>         ;       AX = VBE return status
                        <1>         ;            AX = 004Fh -> succeeded
                        <1>         ;            AX <> 004Fh -> failed
                        <1>         ;
                        <1>         ; Modified registers: eax
                        <1>
00001885 21FF           <1>         and     edi, edi
00001887 7406           <1>         jz      short vbe3_func_fail ; invalid buffer addr !
00001889 EBE0           <1>         jmp     short _vbe_biosfn_return_ctrl_info
                        <1>
                        <1> vbe3_pmfn_return_mode_info:
```

```
                        <1>       ; 21/12/2020
                        <1>       ; 12/12/2020
                        <1>       ;
                        <1>       ; VBE function 4F01h - Return VBE Mode Information
                        <1>       ;
                        <1>       ; Input:
                        <1>       ;     CX = Mode number (VESA VBE mode number)
                        <1>       ;      EDI = Pointer to ModeInfoBlock structure
                        <1>       ;           (256 bytes) -User's buffer address-
                        <1>       ;      EDI = 0 -> kernel call
                        <1>       ;              (do not transfer ModeInfoBlock
                        <1>       ;              to user's buffer address)
                        <1>       ;     AX = 4F01h
                        <1>       ; Output:
                        <1>       ;     AX = VBE return status
                        <1>       ;          AX = 004Fh -> succeeded
                        <1>       ;          AX <> 004Fh -> failed
                        <1>       ;
                        <1>       ; Modified registers: eax, esi, edi
                        <1>
                        <1>       ; int 31h (int 10h) entrance
                        <1>
0000188B 09FF           <1>       or    edi, edi
0000188D 7506           <1>       jnz   short _vbe3_pmfn_return_mode_info
                        <1>
                        <1> vbe3_func_fail:
0000188F B84F010000     <1>       mov   eax, 014Fh ; ah = 1 : Function call failed
                        <1>                        ; al = 4Fh : Function is supported
00001894 C3             <1>       retn
                        <1>
                        <1>       ; jump from '_vbe_biosfn_return_mode_info'
                        <1> _vbe3_pmfn_return_mode_info:
00001895 57             <1>       push  edi
                        <1>
                        <1>       ;; clear vbe3 'mode info block' buffer
                        <1>       ;push  ecx
                        <1>       ;xor   eax, eax
                        <1>       ;mov   ecx, 256/4
                        <1>       ;mov   edi, VBE3MODEINFOBLOCK
                        <1>       ;rep   stosd
                        <1>       ;pop   ecx
                        <1>
                        <1>       ; far call to VESA VBE3 PMI
                        <1>       ;mov   ax, 4F01h ; Return VBE Mode Information
00001896 BF00060000     <1>       mov   edi, VBE3MODEINFOBLOCK-VBE3SAVERESTOREBLOCK
                        <1>       ; ES selector base address = VBE3SAVERESTOREBLOCK
0000189B E8FC000000     <1>       call  int10h_32bit_pmi
                        <1>
000018A0 5F             <1>       pop   edi
                        <1>
                        <1>       ;cmp   ax, 004Fh
                        <1>       ;jne   short vbe3_func_retn  ; failed !
                        <1>
000018A1 21FF           <1>       and   edi, edi
                        <1>       ;jz    short vbe3_func_success
                        <1>       ; 21/12/2020
000018A3 741D           <1>       jz    short vbe3_func_retn
                        <1>
000018A5 6683F84F       <1>       cmp   ax, 004Fh
000018A9 7517           <1>       jne   short vbe3_func_retn  ; failed !
                        <1>
000018AB 51             <1>       push  ecx
000018AC BE007C0900     <1>       mov   esi, VBE3MODEINFOBLOCK
000018B1 B900010000     <1>       mov   ecx, 256
000018B6 E887FC0000     <1>       call  transfer_to_user_buffer
000018BB 59             <1>       pop   ecx
000018BC 72D1           <1>       jc    short vbe3_func_fail
                        <1>
000018BE 31C0           <1>       xor   eax, eax
000018C0 B04F           <1>       mov   al, 4Fh ; successful
                        <1> vbe3_func_success:
                        <1> vbe3_func_retn:
000018C2 C3             <1>       retn
                        <1>
                        <1> vbe3_pmfn_return_current_mode:
                        <1>       ; 12/12/2020
                        <1>       ;
                        <1>       ; VBE function 4F03h - Return Current VBE Mode
                        <1>       ;
                        <1>       ; Input:
                        <1>       ;     none (AX = 4F03h)
                        <1>       ; Output:
                        <1>       ;     AX = VBE return status
                        <1>       ;          AX = 004Fh -> succeeded
                        <1>       ;          AX <> 004Fh -> failed
                        <1>       ;     BX = Current VBE mode
                        <1>       ;       bit 0-13 = Mode number
                        <1>       ;       bit 14 = 0 Windowed frame buffer model
                        <1>       ;              = 1 Linear frame buffer model
                        <1>       ;       bit 15
                        <1>       ;           = 0 Memory cleared at last mode set
                        <1>       ;           = 1 Memory not cleared at last mode set
                        <1>       ;
                        <1>       ; Modified registers: eax, ebx
                        <1>
                        <1>       ; int 31h (int 10h) entrance
                        <1>
                        <1>       ; far call to VESA VBE3 PMI
                        <1>
                        <1>       ;mov   eax, 4F03h ; Return Current VBE Mode
                        <1> vbe3_pmfn_far_call:
                        <1>       ; ES selector base address = VBE3SAVERESTOREBLOCK
                        <1>       ;call  int10h_32bit_pmi
                        <1>       ;retn
000018C3 E9D4000000     <1>       jmp   int10h_32bit_pmi
```

```
                                <1>
                                <1> vbe3_pmfn_set_mode:
                                <1>       ; 22/12/2020
                                <1>       ; 21/12/2020
                                <1>       ; 12/12/2020
                                <1>       ;
                                <1>       ; VBE function 4F02h - Set VBE Mode
                                <1>       ;
                                <1>       ; Input:
                                <1>       ;     BX = Desired Mode to set
                                <1>       ;       bit 0-13 = Mode number
                                <1>       ;       bit 14 = 0 Windowed frame buffer model
                                <1>       ;              = 1 Linear frame buffer model
                                <1>       ;       bit 15
                                <1>       ;           = 0 Memory cleared at last mode set
                                <1>       ;           = 1 Memory not cleared at last mode set
                                <1>       ; Output:
                                <1>       ;     AX = VBE return status
                                <1>       ;          AX = 004Fh -> succeeded
                                <1>       ;          AX <> 004Fh -> failed
                                <1>       ;
                                <1>       ; Modified registers: eax, ebx, esi (21/12/2020)
                                <1>
                                <1>       ; int 31h (int 10h) entrance
                                <1>
                                <1>       ; 22/12/2020 (VESA VBE3 feature)
                                <1>       ; BX bit 11 is flag for
                                <1>       ;     user specified CRTC values for refresh rate
                                <1>       ; 'test bh, 8'
                                <1>       ; if bit 11 is set, EDI points to 'CRTCInfoBlock'
                                <1>
                                <1>       ; 22/12/2020
                                <1>       ;; test bx for VBE video mode
                                <1>       ;test  bh, 1
                                <1>       ;jnz   short vbe3_sm_0
                                <1>
                                <1>       ;; use internal VBE mode set procedure
                                <1>       ;; for non-vbe (std VGA/CGA) modes
                                <1>       ;
                                <1>       ;; (it is useful -as 4F02h function-
                                <1>       ;;  to jump 'vbe_biosfn_set_mode'
                                <1>       ;;  instead of direct jump to '_set_mode')
                                <1>       ;; ((eliminates additional push-pops and settings))
                                <1>
                                <1>       ;jmp   vbe_biosfn_set_mode
                                <1>
                                <1> vbe3_sm_0:
                                <1>       ;;push ds  ; *
                                <1>       ;;push es  ; **
                                <1>       ;;push ebp ; ***
                                <1>       ;;push esi ; ****
                                <1>
                                <1>       ; Fit bx to VESA VBE2 type mode setting
                                <1>       ; (bx bit 11 is used for custom CRTC values in VBE3)
                                <1>       ; clear bit 9 to 11 (clear bh bit 1 to bit 3)
                                <1>
                                <1>       ; 22/12/2020
000018C8 57                     <1>       push   edi ; *****
000018C9 F6C708                 <1>       test   bh, 8  ; Use user specified CRTC values
000018CC 7530                   <1>       jnz    short vbe3_sm_3  ; for refresh rate
                                <1> vbe3_sm_4:
000018CE 80E7C1                 <1>       and    bh, 0C1h ; use bit 15, 14, 8 only (for bh)
                                <1>       ;mov   [vbe_mode_x], bh
                                <1>
000018D1 803D[BA6A0000]03       <1>       cmp    byte [CRT_MODE], 3 ; is current mode 03h ?
000018D8 7509                   <1>       jne    short vbe3_sm_1    ; no
                                <1>
                                <1>       ; save mode 03h video pages and cursor positions
000018DA 57                     <1>       push   edi ; **!***
000018DB 51                     <1>       push   ecx ; ******
                                <1>       ;push  esi
                                <1>
000018DC E8BA040000             <1>       call   save_mode3_multiscreen
                                <1>
                                <1>       ;pop   esi
000018E1 59                     <1>       pop    ecx ; ******
000018E2 5F                     <1>       pop    edi ; **!***
                                <1> vbe3_sm_1:
                                <1>       ; ax = 4F02h
                                <1>       ; bx = video mode number (vbe2 type)
000018E3 E8B4000000             <1>       call   int10h_32bit_pmi
                                <1>                    ; call to far call to VBE3 PMI
                                <1>
000018E8 6683F84F               <1>       cmp    ax, 004Fh ; succeeded ?
000018EC 750E                   <1>       jne    short vbe3_sm_2
                                <1>       ; set current mode byte/sign to extended (SVGA) mode
000018EE C605[BA6A0000]FF       <1>       mov    byte [CRT_MODE], 0FFh ; VESA VBE mode
                                <1>       ; set current VBE mode word to bx input
000018F5 66891D[06120300]       <1>       mov    [video_mode], bx
                                <1> vbe3_sm_2:
                                <1>       ; 22/12/2020
000018FC 5F                     <1>       pop    edi ; *****
000018FD C3                     <1>       retn
                                <1>
                                <1> vbe3_sm_3:
                                <1>       ; 22/12/2020
                                <1>       ; copy user's CRTCInfoBlock to the buffer
000018FE 51                     <1>       push   ecx
000018FF 89FE                   <1>       mov    esi, edi
00001901 BF807D0900             <1>       mov    edi, VBE3CRTCINFOBLOCK
00001906 B940000000             <1>       mov    ecx, 64
0000190B E87CFC0000             <1>       call   transfer_from_user_buffer
00001910 59                     <1>       pop    ecx
                                <1>       ; set offset (es base addr is VBE3SAVERESTOREBLOCK)
00001911 81EF00760900           <1>       sub    edi, VBE3SAVERESTOREBLOCK
```

```
3525 00001917 EBB5                <1>       jmp    short vbe3_sm_4
3526                              <1>
3527                              <1> vesa_vbe3_pmi:
3528                              <1>       ; 12/12/2020
3529                              <1>       ; 08/12/2020
3530                              <1>       ; 07/12/2020
3531                              <1>       ; 05/12/2020, 06/12/2020
3532                              <1>       ; 03/12/2020, 04/12/2020
3533                              <1>       ; 28/11/2020 (TRDOS 386 v2.0.3)
3534                              <1>       ; VGA BIOS functions via
3535                              <1>       ; VESA VBE3 Protected Mode Inface
3536                              <1>       ; [vbe3] = 3 and [pmi32] > 0
3537                              <1>
3538                              <1>       ; 04/12/2020
3539                              <1>       ; Only 'set mode' will be redirected to vbe3 video bios
3540                              <1>       ; (by setting mode 3 multiscreen paraters before and after)
3541                              <1>
3542                              <1>       ; 06/12/2020
3543 00001919 20E4               <1>       and    ah, ah ; 0 = set mode function
3544 0000191B 7402               <1>       jz     short vbe3_pmi_0
3545 0000191D EB76               <1>       jmp    vbe3_pmi_9
3546                              <1>
3547                              <1> vbe3_pmi_0:
3548                              <1>       ; 07/12/2020
3549 0000191F 88C4               <1>       mov    ah, al
3550 00001921 80E480             <1>       and    ah, 80h ; 0 or 80h
3551 00001924 30E0               <1>       xor    al, ah ; 8?h -> 0?h
3552                              <1>
3553                              <1>       ;cmp   al, 13h ; mode number above 13h is returned
3554                              <1>       ;jna   short vbe3_pmi_1
3555                              <1>       ;            ; back to default code due to uncertainty
3556                              <1>       ;            ; (>13h is not std for all svga bioses)
3557                              <1>       ;jmp   VGA_funcs_0
3558                              <1> vbe3_pmi_1:
3559                              <1>       ; 07/12/2020
3560                              <1>       ; Possible cases for VBE3 (PMI, ah=0) set mode:
3561                              <1>       ; current mode > 07h and requested mode: any
3562                              <1>       ; current mode <= 07h and requested mode > 07h
3563                              <1>
3564                              <1>       ; 06/12/2020
3565 00001926 8825[3F8D0100]     <1>       mov    byte [noclearmem], ah ; 0 or 80h
3566                              <1>       ; check current video mode if it is 03h
3567 0000192C 803D[BA6A0000]03   <1>       cmp    byte [CRT_MODE], 3 ; current mode
3568 00001933 750B               <1>       jne    short vbe3_pmi_3
3569                              <1>       ; 07/12/2020
3570                              <1>       ; check new video mode if it is 03h also
3571                              <1>       ;cmp   al, 3
3572                              <1>       ;jne   short vbe3_pmi_2
3573                              <1>       ;mov   byte [p_crt_mode], 80h    ; clear video memory
3574                              <1>       ;jmp   short vbe3_pmi_5
3575                              <1> vbe3_pmi_2:
3576                              <1>       ; case 1:
3577                              <1>       ; Current mode is 03h and new mode is not 03h
3578                              <1>
3579                              <1>       ; save video pages and cursor positions
3580 00001935 56                 <1>       push   esi
3581 00001936 57                 <1>       push   edi
3582 00001937 51                 <1>       push   ecx
3583                              <1>
3584                              <1>       ; 12/12/2020
3585                              <1>       ;mov   esi, 0B8000h ; mode 3 video memory
3586                              <1>       ;mov   edi, 98000h  ; backup location
3587                              <1>       ;mov   ecx, (0B8000h-0B0000h)/4
3588                              <1>       ;rep   movsd
3589                              <1>       ;
3590                              <1>       ;mov   byte [p_crt_mode], 3 ; previous mode, backup sign
3591                              <1>       ;xchg  cl, [ACTIVE_PAGE]
3592                              <1>       ;mov   [p_crt_page], cl  ; save as previous active page
3593                              <1>       ;
3594                              <1>       ;; save cursor positions
3595                              <1>       ;mov   esi, CURSOR_POSN
3596                              <1>       ;mov   edi, cursor_pposn ; cursor positions backup
3597                              <1>       ;mov   cl, 4
3598                              <1>       ;rep   movsd
3599                              <1>
3600                              <1>       ; 12/12/2020
3601 00001938 E85E040000         <1>       call   save_mode3_multiscreen
3602                              <1>
3603 0000193D 59                 <1>       pop    ecx
3604 0000193E 5F                 <1>       pop    edi
3605 0000193F 5E                 <1>       pop    esi
3606                              <1> vbe3_pmi_3:
3607                              <1>       ; 08/12/2020
3608                              <1>       ; 07/12/2020
3609                              <1>       ; case 3 or case 4
3610 00001940 A2[BA6A0000]       <1>       mov    [CRT_MODE], al
3611 00001945 3C03               <1>       cmp    al, 3
3612 00001947 7407               <1>       je     short vbe3_pmi_4
3613                              <1>       ; case 4:
3614                              <1>       ; Current mode is not 03h and also new mode is not 03h
3615 00001949 800D[3D8D0100]80   <1>       or     byte [p_crt_mode], 80h  ; 83h (case 1 -> case 4)
3616                              <1>       ;jmp   short vbe3_pmi_5
3617                              <1> vbe3_pmi_4:
3618                              <1>       ; case 3:
3619                              <1>       ;
3620                              <1>       ; Current mode is not 03h and new mode is 03h
3621                              <1>
3622                              <1> ;vbe3_pmi_5:
3623                              <1>       ;mov   [CRT_MODE], al
3624                              <1>
3625 00001950 E847000000         <1>       call   int10h_32bit_pmi
3626                              <1>
3627 00001955 803D[BA6A0000]03   <1>       cmp    byte [CRT_MODE], 3  ; new video mode
3628                              <1>       ;jne   vbe3_pmi_8   ; video mode <> 03h
3629 0000195C 7532               <1>       jne    short vbe3_pmi_8
```

```
3630                                    <1>
3631                                    <1>     ;push  eax ; 04/12/2020
3632 0000195E 53                        <1>     push   ebx
3633 0000195F 51                        <1>     push   ecx
3634 00001960 52                        <1>     push   edx
3635 00001961 57                        <1>     push   edi ; 03/12/2020
3636                                    <1>
3637                                    <1>     ; 12/12/2020
3638 00001962 56                        <1>     push   esi
3639 00001963 E866040000                <1>     call   restore_mode3_multiscreen
3640 00001968 5E                        <1>     pop    esi
3641                                    <1>     ; AL = active video page
3642                                    <1>
3643                                    <1>     ; 12/12/2020
3644                                    <1>     ;mov   al, [p_crt_page] ; previous mode 3 active page
3645                                    <1>     ;
3646                                    <1>     ;;test byte [p_crt_mode], 7Fh ; 83h or 80h or 03h
3647                                    <1>     ;;jz   short vbe3_pmi_6 ; do not restore video pages
3648                                    <1>     ;                       ; clear current video page
3649                                    <1>     ;; case 3
3650                                    <1>     ;
3651                                    <1>     ;; ([p_crt_mode] = 03h)
3652                                    <1>     ;
3653                                    <1>     ;; New video mode is 3 while current video mode is not 3
3654                                    <1>     ;; (multi screen) video pages will be restored from 098000h
3655                                    <1>     ;
3656                                    <1>     ;; restore video pages and cursor positions
3657                                    <1>     ;
3658                                    <1>     ;mov   [ACTIVE_PAGE], al ; current mode 3 active page
3659                                    <1>     ;
3660                                    <1>     ;push  esi
3661                                    <1>     ;
3662                                    <1>     ;; restore video pages
3663                                    <1>     ;mov   esi, 98000h
3664                                    <1>     ;mov   edi, 0B8000h
3665                                    <1>     ;;mov  ecx, 2000h
3666                                    <1>     ;mov   cx, 2000h ; 8K dwords (32K)
3667                                    <1>     ;rep   movsd
3668                                    <1>     ;
3669                                    <1>     ;mov   [p_crt_mode], cl ; reset ('case 3' end condition)
3670                                    <1>     ;
3671                                    <1>     ;; restore cursor positions
3672                                    <1>     ;mov   esi, cursor_pposn
3673                                    <1>     ;mov   edi, CURSOR_POSN
3674                                    <1>     ;;mov  ecx, 4 ; restore all cursor positions (16 bytes)
3675                                    <1>     ;mov   cl, 4
3676                                    <1>     ;rep   movsd
3677                                    <1>     ;
3678                                    <1>     ;pop   esi
3679                                    <1>     ;
3680                                    <1>     ;; 07/12/2020
3681                                    <1>     ;; restore CRT_START according to ACTIVE_PAGE
3682                                    <1>     ;mov   [CRT_START], cx ; 0
3683                                    <1>     ;
3684                                    <1>     ;; check active page and set it again if it is not 0
3685                                    <1>     ;or    al, al
3686                                    <1>     ;jz    short vbe3_pmi_7
3687                                    <1>     ;
3688                                    <1>     ;mov   cl, al
3689                                    <1> ;vbe3_pmi_5:
3690                                    <1>     ;add   word [CRT_START], 4096
3691                                    <1>     ;dec   cl
3692                                    <1>     ;jnz   short vbe3_pmi_5
3693                                    <1>
3694 00001969 B405                      <1>     mov    ah, 05h ; set current video page
3695                                    <1>     ;al = video page
3696 0000196B E82C000000                <1>     call   int10h_32bit_pmi
3697                                    <1>
3698                                    <1>     ; check current cursor position & set it again if not 0,0
3699                                    <1>     ;movzx ebx, byte [ACTIVE_PAGE]
3700 00001970 0FB6D8                    <1>     movzx  ebx, al
3701 00001973 D0E3                      <1>     shl    bl, 1
3702 00001975 81C3[C6800100]            <1>     add    ebx, CURSOR_POSN
3703 0000197B 668B13                    <1>     mov    dx, [ebx]
3704 0000197E 6621D2                    <1>     and    dx, dx
3705 00001981 7409                      <1>     jz     short vbe3_pmi_7
3706                                    <1>
3707                                    <1>     ;dx = cursor position (dl = column, dh = row)
3708                                    <1>     ;mov   bh, [ACTIVE_PAGE] ; 06/12/2020
3709 00001983 88C7                      <1>     mov    bh, al
3710 00001985 B402                      <1>     mov    ah, 02h ; set cursor position
3711 00001987 E810000000                <1>     call   int10h_32bit_pmi
3712                                    <1>
3713                                    <1>     ;jmp   short vbe3_pmi_7
3714                                    <1>
3715                                    <1> ;vbe3_pmi_6:
3716                                    <1> ;       ; 07/12/2020
3717                                    <1> ;       ; case 1, previous mode is 03h, current mode is 03h
3718                                    <1> ;       ; 03/12/2020
3719                                    <1> ;       cmp    byte [noclearmem], 0
3720                                    <1> ;       jna    short vbe3_pmi_7 ; do not clear memory
3721                                    <1> ;       ; clear video page
3722                                    <1> ;       mov    ecx, 1024 ; 4096/4
3723                                    <1> ;       mov    eax, 07200720h
3724                                    <1> ;       mov    edi, 0B8000h ; [crt_base]
3725                                    <1> ;       add    di, [CRT_START]
3726                                    <1> ;       rep    stosd  ; FILL THE REGEN BUFFER WITH BLANKS
3727                                    <1>
3728                                    <1> vbe3_pmi_7:
3729 0000198C 5F                        <1>     pop    edi
3730 0000198D 5A                        <1>     pop    edx
3731 0000198E 59                        <1>     pop    ecx
3732 0000198F 5B                        <1>     pop    ebx
3733                                    <1>     ;pop   eax ; 04/12/2020
3734                                    <1> vbe3_pmi_8:
```

```
                                <1>       ; 04/12/2020
                                <1>       ;(TRDOS 386 v2.0.3, INT 31h, ah=0 return)
00001990 31C0                   <1>       xor    eax, eax  ; eax = 0 -> succesful
                                <1> vesa_vbe3_pmi_retn:
00001992 07                     <1>       pop    es  ; **
00001993 1F                     <1>       pop    ds  ; *
00001994 CF                     <1>       iretd
                                <1>
                                <1> vbe3_pmi_9:
                                <1>       ; 06/12/2020
                                <1>       ;cmp   ah, 10h ; Set/Get Palette Registers
                                <1>       ;jnb   short vbe3_pmi_10
                                <1>       ; 05/12/2020
00001995 E802000000             <1>       call   int10h_32bit_pmi
0000199A EBF6                   <1>       jmp    short vesa_vbe3_pmi_retn
                                <1>
                                <1> ;vbe3_pmi_10:
                                <1>       ; 06/12/2020
                                <1>       ;jmp   VGA_funcs_0
                                <1>
                                <1> int10h_32bit_pmi:
                                <1>       ; 03/12/2020
                                <1>       ; 28/11/2020
                                <1>       ; calling standard VGA Bios (INT 10h) functions
                                <1>       ; by using 32 bit protected mode interface of
                                <1>       ; VESA VBE3 Video Bios (with 'PMID' signature)
                                <1>
                                <1>       ; 03/12/2020
                                <1>       ; eax, ebx, ecx, edx, edi will be used by vbios pmi
                                <1>       ; (esi and ebp will not be used)
                                <1>
                                <1>       ; 03/12/2020
0000199C 56                     <1>       push   esi
0000199D C1E010                 <1>       shl    eax, 16  ; move function number (ax) to hw
000019A0 8B35[0C120300]         <1>       mov    esi, [pmid_addr] ; linear address of
                                <1>                               ; PMInfo.Entrypoint pointer
                                <1>       ;mov   ax, [esi+PMInfo.EntryPoint]
000019A6 668B06                 <1>       mov    ax, [esi]
000019A9 C1C010                 <1>       rol    eax, 16 ; move PM entry address to hw
                                <1>                      ; and move function number to lw (ax)
000019AC 5E                     <1>       pop    esi
                                <1>
                                <1>       ; top of stack: ; (*)
                                <1>       ; return (the caller) address of "int10h_32bit_pmi"
                                <1>
000019AD E9E7EDFFFF             <1>       jmp    _VBE3PMI_fcall ; will return to the caller (*)
                                <1>
                                <1> _vbe3_pmfn_srs_8:
                                <1>       ; 17/01/2021
000019B2 31DB                   <1>       xor    ebx, ebx ; points to VBE3SAVERESTOREBLOCK
                                <1> _vbe3_pmfn_srs_9: ; 24/01/2021
000019B4 66B8044F               <1>       mov    ax, 4F04h
000019B8 EBE2                   <1>       jmp    short int10h_32bit_pmi
                                <1>
                                <1> vbe3_pmfn_save_restore_state:
                                <1>       ; 24/01/2021
                                <1>       ; 23/01/2021
                                <1>       ; 16/01/2021, 17/01/2021
                                <1>       ; 14/01/2021
                                <1>       ;
                                <1>       ; VBE function 4F04h - Save/Restore Video State
                                <1>       ;
                                <1>       ; Input:
                                <1>       ;      DL = sub function
                                <1>       ;      CL = requested state
                                <1>       ;      EBX = pointer to buffer (if DL<>00h)
                                <1>       ;      AX = 4F04h
                                <1>       ; Output:
                                <1>       ;      AX = 004Fh (successful)
                                <1>       ;      AH > 0 -> error
                                <1>       ;      BX = Number of 64-byte blocks
                                <1>       ;           to hold the state buffer (if DL=00h)
                                <1>
                                <1>       ; Modified registers: eax, ebx, esi, edi
                                <1>
000019BA 21DB                   <1>       and    ebx, ebx ; user's buffer address
000019BC 750A                   <1>       jnz    short _vbe3_pmfn_save_restore_state
                                <1>
000019BE 08D2                   <1>       or     dl, dl
000019C0 740C                   <1>       jz     short _vbe3_pmfn_srs_0
                                <1>
                                <1>       ; function failed
                                <1>       ;;mov  eax, 0100h
                                <1>       ;sub   eax, eax
                                <1>       ;inc   ah  ; eax = 0100h
                                <1>       ;retn
                                <1>       ; 16/01/2021
                                <1> _vbe3_pmfn_srs_fail:
000019C2 B84F010000             <1>       mov    eax, 014Fh ; ah = 1 : Function call failed
                                <1>                         ; al = 4Fh : Function is supported
                                <1> _vbe3_srs_retn:
000019C7 C3                     <1>       retn
                                <1>
                                <1> _vbe3_pmfn_save_restore_state:
000019C8 20D2                   <1>       and    dl, dl
000019CA 7559                   <1>       jnz    short _vbe3_pmfn_srs_2
                                <1> _vbe3_pmfn_srs:
000019CC 31DB                   <1>       xor    ebx, ebx
                                <1> _vbe3_pmfn_srs_0:
                                <1>       ; 24/01/2021
000019CE 83F90F                 <1>       cmp    ecx, 0Fh
                                <1>       ;ja    short _vbe3_pmfn_srs_1
000019D1 77EF                   <1>       ja     short _vbe3_pmfn_srs_fail
                                <1>
                                <1>       ; !!! CLEAR CL BIT 2 !!!
```

```
                                <1>       ; (when bit 2 is set, function causes cpu exception)
                                <1>       ; BIOS data will not be saved and restored
                                <1>       ; (to prevent protected mode page fault error)
000019D3 80E1FD                 <1>       and    cl, ~2 ; and cl, not 2
                                <1>
                                <1>       ; 24/01/2021
                                <1>       ;mov   bl, 1
000019D6 FEC3                   <1>       inc    bl ; = 1
000019D8 66D3E3                 <1>       shl    bx, cl
000019DB 66231D[10120300]       <1>       and    bx, [vbe3stbsflags]
000019E2 7416                   <1>       jz     short _vbe3_pmfn_srs_1
                                <1>       ;mov   bx, cx
000019E4 89CB                   <1>       mov    ebx, ecx ; <= 15
000019E6 D0E3                   <1>       shl    bl, 1 ; 0, 2, 8 .. 30
000019E8 668B9B[BD3B0000]       <1>       mov    bx, [vbestatebufsize+ebx]
000019EF 89DF                   <1>       mov    edi, ebx
                                <1>       ; edi = state buffer size in bytes
000019F1 66C1EB06               <1>       shr    bx, 6 ; / 64
000019F5 66B84F00               <1>       mov    ax, 4Fh
000019F9 C3                     <1>       retn
                                <1> _vbe3_pmfn_srs_1:
                                <1>       ; ax = 4F04h
                                <1>       ;call  int10h_32bit_pmi
                                <1>       ; 24/01/2021
                                <1>       ;call  _vbe3_pmfn_srs_8
                                <1>       ; ebx = 0
000019FA E8B5FFFFFF             <1>       call   _vbe3_pmfn_srs_9
000019FF 6683F84F               <1>       cmp    ax, 004Fh
00001A03 75C2                   <1>       jne    short _vbe3_srs_retn
                                <1>       ; 24/01/2021
                                <1>       ;cmp   ecx, 0Fh
                                <1>       ;ja    short _vbe3_srs_retn
                                <1>       ; 24/01/2021
                                <1>       ;mov   ax, 1
00001A05 B001                   <1>       mov    al, 1
00001A07 66D3E0                 <1>       shl    ax, cl
00001A0A 660905[10120300]       <1>       or     [vbe3stbsflags], ax ; set flag for state option
                                <1>       ; 23/01/2021
00001A11 89DF                   <1>       mov    edi, ebx
00001A13 89C8                   <1>       mov    eax, ecx
00001A15 D0E0                   <1>       shl    al, 1
00001A17 66C1E706               <1>       shl    di, 6 ; * 64
00001A1B 6689B8[BD3B0000]       <1>       mov    [vbestatebufsize+eax], di
                                <1>                           ; save buf size for option
                                <1>       ;xchg  edi, ebx
                                <1>       ; edi = state buffer size in bytes
00001A22 B04F                   <1>       mov    al, 4Fh
00001A24 C3                     <1>       retn
                                <1>
                                <1> _vbe3_pmfn_srs_2:
                                <1>       ; 24/01/2021
                                <1>       ; !!! CLEAR CL BIT 2 !!!
                                <1>       ; (when bit 2 is set, function causes cpu exception)
                                <1>       ; BIOS data will not be saved and restored
                                <1>       ; (to prevent protected mode page fault error)
                                <1>
00001A25 F6C1FD                 <1>       test   cl, ~2 ; test cl, not 2
00001A28 7498                   <1>       jz     short _vbe3_pmfn_srs_fail
                                <1>
00001A2A 80FA02                 <1>       cmp    dl, 2
00001A2D 7748                   <1>       ja     short _vbe3_pmfn_srs_5
                                <1>
                                <1>       ;and   cl, ~2 ; and cl, not 2
                                <1>
00001A2F 53                     <1>       push   ebx ; * ; buffer address
                                <1>       ; save or restore state
                                <1>       ; (get required buffer size at first)
00001A30 52                     <1>       push   edx ; **
00001A31 28D2                   <1>       sub    dl, dl ; 0
00001A33 E894FFFFFF             <1>       call   _vbe3_pmfn_srs
00001A38 5A                     <1>       pop    edx ; **
                                <1>       ; 24/01/2021
00001A39 5B                     <1>       pop    ebx ; *
00001A3A 08E4                   <1>       or     ah, ah
00001A3C 7538                   <1>       jnz    short _vbe3_pmfn_srs_4 ; error
                                <1>
                                <1>       ; edi = buffer size in bytes
00001A3E 81FF00080000           <1>       cmp    edi, 2048
00001A44 772B                   <1>       ja     short _vbe3_pmfn_srs_3
                                <1>
00001A46 80FA01                 <1>       cmp    dl, 1
00001A49 7531                   <1>       jne    short _vbe3_pmfn_srs_6 ; restore state
                                <1>
                                <1>       ; save video state
                                <1>       ;xor   ebx, ebx ; points to VBE3SAVERESTOREBLOCK
                                <1>       ;mov   ax, 4F04h
                                <1>       ;call  int10h_32bit_pmi
                                <1>
                                <1>       ; 24/01/2021
00001A4B E842000000             <1>       call   _vbe3_pmfn_srs_7
                                <1>
00001A50 6683F84F               <1>       cmp    ax, 004Fh
00001A54 7520                   <1>       jne    short _vbe3_pmfn_srs_4
                                <1>
00001A56 09DB                   <1>       or     ebx, ebx  ; kernel ('sysvideo') ?
00001A58 741C                   <1>       jz     short _vbe3_pmfn_srs_4 ; yes
                                <1>
                                <1>       ; the caller is user
00001A5A 51                     <1>       push   ecx ; *
00001A5B 89F9                   <1>       mov    ecx, edi ; state buffer size
00001A5D BE00760900             <1>       mov    esi, VBE3SAVERESTOREBLOCK ; source
                                <1>                                 ; (vbe3 pmi buff)
00001A62 89DF                   <1>       mov    edi, ebx     ; destination (user buff)
00001A64 E8D9FA0000             <1>       call   transfer_to_user_buffer
00001A69 59                     <1>       pop    ecx ; *
```

```
3945 00001A6A 7205              <1>       jc     short _vbe3_pmfn_srs_3
3946                            <1>
3947 00001A6C 29C0              <1>       sub    eax, eax
3948 00001A6E B04F              <1>       mov    al, 4Fh
3949 00001A70 C3                <1>       retn
3950                            <1>
3951                            <1>       ; 24/01/2021
3952                            <1> _vbe3_pmfn_srs_3:
3953 00001A71 B84F010000        <1>       mov    eax, 014Fh
3954                            <1> _vbe3_pmfn_srs_4:
3955 00001A76 C3                <1>       retn
3956                            <1> _vbe3_pmfn_srs_5:
3957 00001A77 31C0              <1>       xor    eax, eax
3958 00001A79 FEC4              <1>       inc    ah
3959                            <1>       ; eax = 0100h, function is not supported
3960 00001A7B C3                <1>       retn
3961                            <1>
3962                            <1> _vbe3_pmfn_srs_6:
3963                            <1>       ; restore video state
3964                            <1>       ; 24/01/2021
3965                            <1>       ;pop   ebx ; *
3966                            <1>       ; 23/01/2021
3967 00001A7C 09DB              <1>       or     ebx, ebx ; 0 ?
3968 00001A7E 7412              <1>       jz     short _vbe3_pmfn_srs_7 ; 'sysvideo' call
3969                            <1>       ; 24/01/2021
3970                            <1>       ;jz    _vbe3_pmfn_srs_8
3971 00001A80 89DE              <1>       mov    esi, ebx
3972                            <1>       ; esi = user's video state buffer
3973 00001A82 51                <1>       push   ecx ; *
3974 00001A83 89F9              <1>       mov    ecx, edi ; state buffer size
3975 00001A85 BF00760900        <1>       mov    edi, VBE3SAVERESTOREBLOCK ; destination
3976                            <1>                                     ; (vbe3 pmi buff)
3977                            <1>       ;mov   esi, ebx     ; source (user buff)
3978 00001A8A E8FDFA0000        <1>       call   transfer_from_user_buffer
3979 00001A8F 59                <1>       pop    ecx ; *
3980 00001A90 72DF              <1>       jc     short _vbe3_pmfn_srs_3
3981                            <1> _vbe3_pmfn_srs_7:
3982 00001A92 53                <1>       push   ebx ; *
3983                            <1>       ; restore video state
3984                            <1>       ;xor   ebx, ebx ; points to VBE3SAVERESTOREBLOCK
3985                            <1>       ;mov   ax, 4F04h
3986                            <1>       ;call  int10h_32bit_pmi
3987                            <1>       ; 17/01/2021
3988 00001A93 E81AFFFFFF        <1>       call   _vbe3_pmfn_srs_8
3989 00001A98 5B                <1>       pop    ebx ; *
3990 00001A99 C3                <1>       retn
3991                            <1>
3992                            <1> VIDEO_STATE:
3993                            <1>       ; 26/06/2016
3994                            <1>       ; 12/05/2016
3995                            <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
3996                            <1>
3997                            <1> ;-------------------------------------------------
3998                            <1> ; VIDEO STATE
3999                            <1> ;  RETURNS THE CURRENT VIDEO STATE IN AX
4000                            <1> ;  AH = NUMBER OF COLUMNS ON THE SCREEN
4001                            <1> ;  AL = CURRENT VIDEO MODE
4002                            <1> ;  BH = CURRENT ACTIVE PAGE
4003                            <1> ;-------------------------------------------------
4004                            <1>
4005 00001A9A 8A25[BC6A0000]    <1>       mov    ah, [CRT_COLS]      ; GET NUMBER OF COLUMNS
4006 00001AA0 A0[BA6A0000]      <1>       mov    al, [CRT_MODE]      ; CURRENT MODE
4007                            <1>       ;movzx esi, al
4008                            <1>       ;mov   ah, [esi+M6]
4009                            <1>       ; BH = active page
4010 00001AA5 8A3D[D6800100]    <1>       mov    bh, [ACTIVE_PAGE] ; GET CURRENT ACTIVE PAGE
4011 00001AAB FA                <1>       cli    ; 02/01/2017
4012 00001AAC 5D                <1>       pop    ebp          ; RECOVER REGISTERS
4013 00001AAD 5F                <1>       pop    edi
4014 00001AAE 5E                <1>       pop    esi
4015 00001AAF 59                <1>       pop    ecx          ; DISCARD SAVED BX
4016 00001AB0 EB41              <1>       jmp    short M15    ; RETURN TO CALLER
4017                            <1>
4018                            <1> set_mode_ncm:
4019                            <1>       ; 17/11/2020 (TRDOS 386 v2.0.3)
4020                            <1>       ; 04/07/2016 - TRDOS 386 (TRDOS v2.0)
4021                            <1>       ; set mode without clearing the video memory
4022                            <1>       ; (ony for graphics modes)
4023                            <1>
4024                            <1>       ;cmp   al, 7 ; IBM PC CGA modes
4025                            <1>       ;jna   short SET_MODE ; normal function (clear)
4026                            <1>       ;; do not clear memory
4027                            <1>       ;;mov  [noclearmem], al ; > 0
4028                            <1>       ;mov   byte [noclearmem], 80h ; 17/11/2020
4029                            <1>       ;call  _set_mode
4030                            <1>       ;mov   byte [noclearmem], 0
4031                            <1>         ;jmp     short VIDEO_RETURN
4032                            <1>
4033                            <1>       ; 17/11/2020 (TRDOS v2.0.3)
4034 00001AB2 0C80              <1>       or     al, 80h ; not clear memory option
4035                            <1>
4036                            <1>       ; 05/12/2020
4037                            <1>       ; 27/11/2020
4038                            <1>       ; 17/11/2020
4039                            <1>       ; 08/08/2016, 10/08/2016
4040                            <1>       ; 29/07/2016, 30/07/2016
4041                            <1>       ; 25/07/2016, 26/07/2016, 27/07/2016
4042                            <1>       ; 02/07/2016, 18/07/2016, 23/07/2016
4043                            <1>       ; 24/06/2016, 26/06/2016
4044                            <1>       ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
4045                            <1> SET_MODE:
4046                            <1>       ; For 32 bit TRDOS and Retro UNIX 386:
4047                            <1>       ;      valid video mode: 03h only!
4048                            <1>       ;      (VGA modes will be selected with another routine)
4049                            <1>       ;
```

```
4050                                    <1>       ; set_txt_mode ; 80*25 (16 fore colors, 8 back colors)
4051                                    <1>
4052                                    <1>       ; 27/11/2020
4053                                    <1>
4054                                    <1>       ; Check if current mode is
4055                                    <1>       ; Bochs/Plex86 VBE graphics mode
4056 00001AB4 803D[BA6A0000]FF         <1>       cmp    byte [CRT_MODE], 0FFh ; VESA VBE graphics mode
4057 00001ABB 7220                     <1>       jb     short _set_mode_      ; signature
4058                                    <1>                                  ; VBE mode number is in
4059                                    <1>                                  ; [video_mode] bit 0to8
4060 00001ABD 88C3                     <1>       mov    bl, al ; save video mode
4061 00001ABF E84D240000               <1>       call   dispi_get_enable
4062 00001AC4 50                       <1>       push   eax ; save current VBE dispi status
4063                                    <1>       ; Disable Bochs/Plex86 VBE dispi
4064                                    <1>       ;mov   ax, 0 ; VBE_DISPI_DISABLED
4065 00001AC5 31C0                     <1>       xor    eax, eax ; 0
4066 00001AC7 E819240000               <1>       call   dispi_set_enable
4067 00001ACC 88D8                     <1>       mov    al, bl ; restore video mode
4068 00001ACE E827000000               <1>       call   _set_mode
4069 00001AD3 58                       <1>       pop    eax ; restore current VBE dispi status
4070 00001AD4 7313                     <1>       jnc    short VIDEO_RETURN
4071                                    <1>       ; ! unimplemented or invalid video mode number !
4072                                    <1>       ; VBE dispi must be enabled again
4073                                    <1>       ; (return to run on current VBE graphics mode)
4074                                    <1>       ;;mov  al, [video_mode+1] ; bit 8 to 15
4075                                    <1>       ;;and  al, 0C0h ; isolate bit 14 and bit 15
4076                                    <1>       ;;or   al, 1 ; VBE_DISPI_ENABLED
4077 00001AD6 E80A240000               <1>       call   dispi_set_enable
4078 00001ADB EB07                     <1>       jmp    short _video_func_err
4079                                    <1>
4080                                    <1> _set_mode_:
4081                                    <1>       ; VGA bios (non-VBE) 'setmode' procedure
4082                                    <1>
4083                                    <1>       ; 26/11/2020 (TRDOS v2.0.3)
4084                                    <1>
4085                                    <1> ;--------------------------------------------------
4086                                    <1> ; SET MODE                                      :
4087                                    <1> ;     THIS ROUTINE INITIALIZES THE ATTACHMENT TO    :
4088                                    <1> ;     THE SELECTED MODE, THE SCREEN IS BLANKED.     :
4089                                    <1> ; INPUT                                          :
4090                                    <1> ;     (AL) - MODE SELECTED (RANGE 0-7)       :
4091                                    <1> ; OUTPUT                                        :
4092                                    <1> ;     NONE                                      :
4093                                    <1> ;--------------------------------------------------
4094                                    <1>
4095 00001ADD E818000000               <1>       call   _set_mode ; 24/06/2016 (set_txt_mode)
4096                                    <1>       ; 26/11/2020
4097 00001AE2 7305                     <1>       jnc    short VIDEO_RETURN
4098                                    <1>
4099                                    <1>       ; 26/11/2020
4100                                    <1> _video_func_err:
4101 00001AE4 31C0                     <1>       xor    eax, eax ; function call failed
4102 00001AE6 48                       <1>       dec    eax  ; 0FFFFFFFFh ; - 1
4103 00001AE7 EB05                     <1>       jmp    short _video_return
4104                                    <1>
4105                                    <1> ; 12/05/2016
4106                                    <1> ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
4107                                    <1>
4108                                    <1> ;-----      NORMAL RETURN FROM ALL VIDEO RETURNS
4109                                    <1>
4110                                    <1> VIDEO_RETURN:
4111 00001AE9 A1[308D0100]             <1>       mov    eax, [video_eax] ; 12/05/2016
4112                                    <1> _video_return:
4113 00001AEE FA                       <1>       cli ; 02/01/2017
4114 00001AEF 5D                       <1>       pop    ebp ; ******** ; 26/11/2020
4115 00001AF0 5F                       <1>       pop    edi ; *******
4116 00001AF1 5E                       <1>       pop    esi ; ******
4117 00001AF2 5B                       <1>       pop    ebx ; *****
4118                                    <1> M15:  ; VIDEO_RETURN_C
4119                                    <1>       ;;15/01/2017
4120                                    <1>       ; 02/01/2017
4121                                    <1>       ;;mov  byte [intflg], 0
4122                                    <1>       ;
4123 00001AF3 59                       <1>       pop    ecx ; **** ; 26/11/2020
4124 00001AF4 5A                       <1>       pop    edx ; ***
4125 00001AF5 1F                       <1>       pop    ds  ; **
4126 00001AF6 07                       <1>       pop    es  ; *      ; RECOVER SEGMENTS
4127 00001AF7 CF                       <1>       iretd        ; ALL DONE
4128                                    <1>
4129                                    <1> set_txt_mode:
4130                                    <1>
4131                                    <1>       ; 29/07/2016
4132                                    <1>       ; 27/06/2016
4133 00001AF8 B003                     <1>       mov    al, 3 ; 26/11/2020 (bit 7 = 0)
4134                                    <1>
4135                                    <1>       ; 17/11/2020 (TRDOS v2.0.3)
4136                                    <1>       ;mov   byte [noclearmem], 0
4137                                    <1>
4138                                    <1> ; 12/04/2021
4139                                    <1> ; 10/08/2016
4140                                    <1> ; 08/08/2016
4141                                    <1> ; 30/07/2016
4142                                    <1> ; 29/07/2016
4143                                    <1> ; 25/07/2016, 26/07/2016, 27/07/2016
4144                                    <1> ; 07/07/2016, 18/07/2016, 23/07/2016
4145                                    <1> ; 02/07/2016, 03/07/2016, 04/07/2016
4146                                    <1> ; 26/06/2016
4147                                    <1> ; 24/06/2016 (set_txt_mode -> _set_mode)
4148                                    <1> ; 17/06/2016
4149                                    <1> ; 29/05/2016
4150                                    <1> ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
4151                                    <1>
4152                                    <1> _set_mode:
4153                                    <1>       ; 12/12/2020
4154                                    <1>       ; 26/11/2020
```

```
4155                                    <1>     ; call from 'biosfn_set_video_mode'
4156                                    <1>     ;      (bochs/plex86 video bios code)
4157                                    <1>     ; call from 'SET_MODE'
4158                                    <1>     ;      (TRDOS 386 v2 default, IBM PC/AT rom bios code)
4159                                    <1>     ; continue from 'set_txt_mode'
4160                                    <1>
4161                                    <1>     ; INPUT:
4162                                    <1>     ;      al = VGA video mode
4163                                    <1>     ; RETURN:
4164                                    <1>     ;      cf = 1 -> video mode not implemented
4165                                    <1>     ;      cf = 0 -> OK
4166                                    <1>     ;
4167                                    <1>     ; Modified registers: eax, bx, ecx, esi, edi, (ebp)
4168                                    <1>
4169                                    <1>     ; 17/11/2020 (TRDOS v2.0.3)
4170                                    <1>     ; no clear memory option
4171                                    <1>     ; (from mode number byte bit 7)
4172 00001AFA 88C4                     <1>     mov    ah, al
4173 00001AFC 80E480                   <1>     and    ah, 80h
4174                                    <1>     ;mov   [noclearmem], al
4175 00001AFF 8825[3F8D0100]           <1>     mov    [noclearmem], ah
4176                                    <1>     ;and   al, 7Fh ; clear bit 7
4177                                    <1>     ;;xor  [noclearmem], al ; clear bit 0 to 6
4178                                    <1>     ; 26/11/2020
4179 00001B05 30E0                     <1>     xor    al, ah ; and al, 7Fh
4180                                    <1>
4181                                    <1>     ; 19/11/2020
4182                                    <1>
4183                                    <1>     ; Video mode 03h action principle:
4184                                    <1>     ;
4185                                    <1>     ; for case 1:
4186                                    <1>     ; Current mode is 03h and next/requested mode is not 03h
4187                                    <1>     ;      - save mode (set mode 03h flag)
4188                                    <1>     ;      - save 8 video pages (which are will be restored)
4189                                    <1>     ;      - save active page number (which will be reactivated)
4190                                    <1>     ;      - set active page to 0 always (no multi screen)
4191                                    <1>     ;      - save 8 cursor positions (which will be restored)
4192                                    <1>     ;      - use 'noclearmem' option
4193                                    <1>     ;      [p_crt_mode] = 0 -> 03h
4194                                    <1>     ;
4195                                    <1>     ; for case 2:
4196                                    <1>     ; Current mode is 03h and next/requested mode is also 03h
4197                                    <1>     ;      - clear active video page if 'noclearmem' is not set
4198                                    <1>     ;      [p_crt_mode] = 0 -> 80h -> 0
4199                                    <1>     ;
4200                                    <1>     ; for case 3:
4201                                    <1>     ; Current mode is not 03h and next/requested mode is 03h
4202                                    <1>     ;      - restore video pages (8 video pages were saved)
4203                                    <1>     ;      - restore active page number (which were saved)
4204                                    <1>     ;      - restore 8 cursor positions (which were saved)
4205                                    <1>     ;      - reset/clear mode 03h flag
4206                                    <1>     ;      [p_crt_mode] = 03h -> 0
4207                                    <1>     ;
4208                                    <1>     ; for case 4:
4209                                    <1>     ; Current mode is not 03h and next/requested mode is not 03h
4210                                    <1>     ;      - use 'noclearmem' option
4211                                    <1>     ;      - set active page to 0 always
4212                                    <1>     ;      [p_crt_mode] = 03h -> 83h -> 03h
4213                                    <1>     ;
4214                                    <1>     ; initial (boot time) values:
4215                                    <1>     ;      [p_crt_mode] = 0 ("there isn't a page backup, yet")
4216                                    <1>     ;        [CRT_MODE] = 3 (kernel's starting mode)
4217                                    <1>
4218                                    <1>     ; 26/11/2020
4219 00001B07 3C03                     <1>     cmp    al, 03h          ; mode 3, 80x25 text, 16 colors
4220 00001B09 7515                     <1>     jne    short _sm_0 ; (default mode for TRDOS 386 mainprog)
4221                                    <1>
4222                                    <1>     ; case 2 or case 3
4223                                    <1>
4224                                    <1>     ; check current video mode if it is 03h
4225 00001B0B 08E4                     <1>     or     ah, ah ; 80h or 0 ('noclearmem' option)
4226 00001B0D 7521                     <1>     jnz    short _sm_1 ; do not clear display page
4227                                    <1>
4228                                    <1>     ; 26/11/2020
4229                                    <1>     ; Note:
4230                                    <1>     ; [CRT_MODE] = 0FFh for VESA VBE video modes
4231                                    <1>     ; [video_mode] = standard VGA and VESA VBE video modes
4232                                    <1>
4233 00001B0F 3805[BA6A0000]           <1>     cmp    [CRT_MODE], al      ; 03h
4234 00001B15 7520                     <1>     jne    short _sm_2  ; case 3 ([p_crt_mode] = 03h)
4235                                    <1>
4236                                    <1>     ; case 2
4237                                    <1>
4238                                    <1>     ; [p_crt_mode] = 0
4239                                    <1>
4240                                    <1>     ; 19/11/2020
4241                                    <1>     ; If '_set_mode' procedure is called for video mode 3
4242                                    <1>     ;     while video mode is 3, video page will be cleared
4243                                    <1>     ;     and cursor position of video page will be reset.
4244                                    <1>
4245                                    <1>     ; clear display page
4246 00001B17 C605[3D8D0100]80          <1>     mov    byte [p_crt_mode], 80h ; clear page sign
4247 00001B1E EB1C                     <1>     jmp    short _sm_3  ; bypass save video page routine
4248                                    <1> _sm_0:
4249                                    <1>     ; case 1 or case 4
4250                                    <1>
4251                                    <1>     ; 05/12/2020
4252 00001B20 803D[BA6A0000]03         <1>     cmp    byte [CRT_MODE], 3 ; is current mode 03h?
4253 00001B27 7507                     <1>     jne    short _sm_1  ; case 4 ; [p_crt_mode] = 03h
4254                                    <1>
4255                                    <1>     ; case 1
4256                                    <1>     ; [p_crt_mode] = 0
4257                                    <1>
4258                                    <1>     ; 19/11/2020
4259                                    <1>     ; If '_set_mode' procedure is called for a video mode
```

```
4260                            <1>       ;     except video mode 3 while current video mode
4261                            <1>       ;     is 3, all video pages of mode 3 will be copied
4262                            <1>       ;     to 98000h address as backup, before mode change.
4263                            <1>
4264                            <1> _sm_save_pm:
4265                            <1>       ; 12/12/2020
4266                            <1>       ;; 03/07/2016
4267                            <1>       ;; save video pages
4268                            <1>       ;mov   esi, 0B8000h
4269                            <1>       ;mov   edi, 98000h ; 30/07/2016
4270                            <1>       ;mov   ecx, (0B8000h-0B0000h)/4
4271                            <1>       ;rep   movsd
4272                            <1>
4273                            <1>       ;mov   byte [p_crt_mode], 3 ; previous mode, backup sign
4274                            <1>       ;; 26/11/2020
4275                            <1>       ;xchg  cl, [ACTIVE_PAGE]
4276                            <1>       ;mov   [p_crt_page], cl  ; save as previous active page
4277                            <1>       ;
4278                            <1>       ;; save cursor positions
4279                            <1>       ;mov   esi, CURSOR_POSN
4280                            <1>       ;mov   edi, cursor_pposn ; cursor positions backup
4281                            <1>       ;mov   cl, 4
4282                            <1>       ;rep   movsd
4283                            <1>
4284                            <1>       ; 12/12/2020
4285 00001B29 E86D020000       <1>       call   save_mode3_multiscreen
4286                            <1>
4287                            <1>       ; 29/07/2016
4288                            <1>       ;;mov  [ACTIVE_PAGE], cl ; 0
4289                            <1>       ;xchg  cl, [ACTIVE_PAGE]
4290                            <1>       ;mov   [p_crt_page], cl  ; previous page (for mode 3)
4291                            <1>
4292                            <1>       ; [ACTIVE_PAGE] = 0
4293                            <1>
4294 00001B2E EB07             <1>       jmp    short _sm_2  ; case 1 - 19/11/2020
4295                            <1> _sm_1:
4296                            <1>       ; 26/11/2020
4297                            <1>       ; 19/11/2020
4298                            <1>
4299                            <1>       ; case 4
4300 00001B30 800D[3D8D0100]80  <1>       or     byte [p_crt_mode], 80h
4301                            <1>                           ; here [p_crt_mode] must be 83h
4302                            <1>                           ;       (for case 4)
4303                            <1>                           ; (because video mode 03h
4304                            <1>                           ; was changed before as in case 1)
4305                            <1>
4306                            <1> _sm_2:       ; case 4 (jump to _sm_2) - 19/11/2020
4307                            <1>
4308                            <1>       ; 19/11/2020
4309                            <1>       ; case 3
4310                            <1>       ; If '_set_mode' procedure is called for video mode 3
4311                            <1>       ;     while video mode is not 3 and if there is video
4312                            <1>       ;     page backup for video mode 3, all (of 8) mode 3
4313                            <1>       ;     video pages will be restored from 98000h.
4314                            <1>
4315 00001B37 A2[BA6A0000]     <1>       mov    [CRT_MODE], al  ; save mode in global variable
4316                            <1> _sm_3:
4317                            <1>       ; 30/07/2016
4318                            <1>       ; 26/07/2016
4319                            <1>       ; 25/07/2016
4320                            <1>       ; set_mode_vga:
4321                            <1>       ; 18/07/2016
4322                            <1>       ; 14/07/2016
4323                            <1>       ; 09/07/2016
4324                            <1>       ; 04/07/2016
4325                            <1>       ; 03/07/2016 (TRDOS 386 = TRDOS v2.0)
4326                            <1>       ; /// video mode 13h ///
4327                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
4328                            <1>       ; vgabios-0.7a (2011)
4329                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
4330                            <1>       ; 'vgabios.c', 'vgatables.h'
4331                            <1>       ;
4332                            <1>       ; Oracle VirtualBox 5.0.24 VGABios Source Code
4333                            <1>       ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
4334                            <1>       ;
4335 00001B3C 88C4               <1>       mov    ah, al
4336 00001B3E B910000000       <1>       mov    ecx, vga_mode_count
4337 00001B43 BE[D66A0000]     <1>       mov    esi, vga_modes
4338 00001B48 31DB             <1>       xor    ebx, ebx
4339                            <1> _sm_4:
4340 00001B4A AC               <1>       lodsb
4341 00001B4B 38C4             <1>       cmp    ah, al
4342 00001B4D 7406             <1>       je     short _sm_5
4343 00001B4F FEC3             <1>       inc    bl
4344 00001B51 E2F7             <1>       loop   _sm_4
4345                            <1>
4346                            <1>       ; UNIMPLEMENTED VIDEO MODE !
4347                            <1>       ;xor   eax, eax
4348                            <1>         ;mov[video_eax], eax ; 0
4349                            <1>
4350                            <1>       ; 26/11/2020
4351 00001B53 F9               <1>       stc    ; unimplemented video mode ! (cf=1)
4352                            <1>
4353 00001B54 C3               <1>       retn
4354                            <1>
4355                            <1> ;-----       eBX POINTS TO CORRECT ROW OF INITIALIZATION TABLE
4356                            <1>
4357                            <1> _sm_5:       ; 25/07/2016
4358                            <1>       ;mov   esi, ebx
4359                            <1>       ;add   esi, vga_memmodel
4360                            <1>       ;mov   al, [esi]
4361                            <1>       ; 19/11/2020
4362 00001B55 8A83[266B0000]    <1>       mov    al, [ebx+vga_memmodel]
4363 00001B5B A2[568D0100]      <1>       mov    [VGA_MTYPE], al
4364                            <1>
```

```
4365 00001B60 89DF              <1>       mov   edi, ebx
4366 00001B62 81C7[366B0000]    <1>       add   edi, vga_dac_s
4367 00001B68 C0E302            <1>       shl   bl, 2 ; byte -> dword
4368 00001B6B 81C3[E66A0000]    <1>       add   ebx, vga_mode_tbl_ptr
4369                            <1>
4370                            <1>       ;mov  dword [VGA_BASE], 0B8000h
4371                            <1>       ;cmp  ah, 0Dh ; [CRT_MODE]
4372                            <1>       ;jb   short M9
4373                            <1>       ;mov  dword [VGA_BASE], 0A0000h
4374                            <1> ;M9:
4375 00001B71 8B33              <1>       mov   esi, [ebx]
4376 00001B73 89F3              <1>       mov   ebx, esi
4377 00001B75 83C614            <1>       add   esi, vga_p_cm_pos ; ebx + 20
4378 00001B78 668B06            <1>       mov   ax, [esi]       ; get the cursor mode from the table
4379 00001B7B 66A3[D36A0000]    <1>       mov   [CURSOR_MODE], ax ; save cursor mode (initial value)
4380                            <1>       ; al = 6, ah = 7
4381                            <1>       ; al = 0Dh, ah = 0Eh ; 25/07/2016
4382 00001B81 E894020000        <1>       call  cursor_shape_fix
4383                            <1>       ; al = 14, ah = 15 (If [CHAR_HEIGHT] = 16)
4384 00001B86 668906            <1>       mov   [esi], ax
4385                            <1>
4386 00001B89 56                <1>       push  esi ; *
4387                            <1>
4388                            <1>       ; 17/04/2021
4389 00001B8A B603              <1>       mov   dh, 03h
4390                            <1>       ;
4391 00001B8C 8A25[C16A0000]    <1>       mov   ah, [VGA_MODESET_CTL]
4392 00001B92 80E408            <1>       and   ah, 8 ; default palette loading ?
4393 00001B95 7522              <1>       jnz   short _sm_6
4394                            <1>       ;mov  dx, 3C6h ; VGAREG_PEL_MASK (DAC mask register)
4395                            <1>       ; 17/04/2021
4396 00001B97 B2C6              <1>       mov   dl, 0C6h
4397 00001B99 B0FF              <1>       mov   al, 0FFh ; PEL mask
4398 00001B9B EE                <1>       out   dx, al
4399 00001B9C 8A27              <1>       mov   ah, [edi] ; DAC model (selection number)
4400 00001B9E E840100000        <1>       call  load_dac_palette
4401                            <1>       ; ecx = 0
4402 00001BA3 F605[C16A0000]02  <1>       test  byte [VGA_MODESET_CTL], 2 ; gray scale summing
4403 00001BAA 740D              <1>       jz    short _sm_6
4404 00001BAC 53                <1>       push  ebx
4405 00001BAD 29DB              <1>       sub   ebx, ebx ; sub bl, bl
4406 00001BAF 66B90001          <1>         mov     cx, 256
4407 00001BB3 E87E100000        <1>       call  gray_scale_summing
4408 00001BB8 5B                <1>       pop   ebx
4409                            <1> _sm_6:
4410                            <1>       ; Reset Attribute Ctl flip-flop
4411                            <1>       ;mov  dx, 3DAh ; VGAREG_ACTL_RESET
4412                            <1>       ; 17/03/2021
4413 00001BB9 B2DA              <1>       mov   dl, 0DAh ; dx = 3DAh
4414 00001BBB EC                <1>       in    al, dx
4415                            <1>       ; Set Attribute Ctl
4416 00001BBC 89DE              <1>       mov   esi, ebx ; addr of params tbl for selected mode
4417 00001BBE 83C623            <1>       add   esi, 35  ; actl regs
4418 00001BC1 30E4              <1>       xor   ah, ah ; 0
4419                            <1>       ;mov  dx, 3C0h ; VGAREG_ACTL_ADDRESS
4420                            <1>       ; 17/04/2021
4421 00001BC3 B2C0              <1>       mov   dl, 0C0h
4422                            <1> _sm_7:
4423 00001BC5 88E0              <1>       mov   al, ah
4424 00001BC7 EE                <1>       out   dx, al ; index
4425 00001BC8 AC                <1>       lodsb
4426                            <1>       ; DX = 3C0h = VGAREG_ACTL_WRITE_DATA
4427 00001BC9 EE                <1>       out   dx, al ; value
4428 00001BCA FEC4              <1>       inc   ah
4429 00001BCC 80FC14            <1>       cmp   ah, 20 ; number of actl registers
4430 00001BCF 72F4              <1>       jb    short _sm_7
4431                            <1>       ;
4432 00001BD1 88E0              <1>       mov   al, ah ; 20
4433 00001BD3 EE                <1>       out   dx, al ; index
4434 00001BD4 28C0              <1>       sub   al, al ; 0
4435 00001BD6 EE                <1>       out   dx, al ; value
4436                            <1>       ;
4437                            <1>       ; Set Sequencer Ctl
4438 00001BD7 89DE              <1>       mov   esi, ebx ; addr of params tbl for selected mode
4439 00001BD9 83C605            <1>       add   esi, 5 ; sequ regs
4440                            <1>       ;
4441                            <1>       ;mov  dx, 3C4h  ; VGAREG_SEQU_ADDRESS
4442                            <1>       ; 17/04/2021
4443 00001BDC B2C4              <1>       mov   dl, 0C4h
4444 00001BDE EE                <1>       out   dx, al ; 0
4445                            <1>       ;inc  dx ; 3C5h ; VGAREG_SEQU_DATA
4446                            <1>       ; 17/04/2021
4447 00001BDF FEC2              <1>       inc   dl ; dx = 3C5h
4448 00001BE1 B003              <1>       mov   al, 3
4449 00001BE3 EE                <1>       out   dx, al
4450 00001BE4 B401              <1>       mov   ah, 1
4451                            <1> _sm_8:
4452 00001BE6 88E0              <1>       mov   al, ah
4453                            <1>       ;mov  dx, 3C4h ; VGAREG_SEQU_ADDRESS
4454                            <1>       ;dec  dx
4455                            <1>       ; 17/04/2021
4456 00001BE8 FECA              <1>       dec   dl ; dx = 3C4h
4457 00001BEA EE                <1>       out   dx, al ; index
4458 00001BEB AC                <1>       lodsb
4459                            <1>       ;inc  dx ; 3C5h ; VGAREG_SEQU_DATA
4460                            <1>       ; 17/04/2021
4461 00001BEC FEC2              <1>       inc   dl
4462 00001BEE EE                <1>       out   dx, al
4463 00001BEF 80FC04            <1>       cmp   ah, 4 ; number of sequ regs
4464 00001BF2 7304              <1>       jnb   short _sm_9
4465 00001BF4 FEC4              <1>       inc   ah
4466 00001BF6 EBEE              <1>       jmp   short _sm_8
4467                            <1> _sm_9:
4468                            <1>       ; Set Grafx Ctl
4469 00001BF8 89DE              <1>       mov   esi, ebx ; addr of params tbl for selected mode
```

```
4470 00001BFA 83C637              <1>       add   esi, 55 ; grdc regs
4471 00001BFD 30E4                <1>       xor   ah, ah ; 0
4472                              <1> _sm_10:
4473 00001BFF 88E0                <1>       mov   al, ah
4474                              <1>       ;mov  dx, 3CEh ; VGAREG_GRDC_ADDRESS
4475                              <1>       ; 17/04/2021
4476 00001C01 B2CE                <1>       mov   dl, 0CEh
4477 00001C03 EE                  <1>       out   dx, al
4478 00001C04 AC                  <1>       lodsb
4479                              <1>       ;inc  dx ; 3CFh ; VGAREG_GRDC_DATA
4480                              <1>       ; 17/04/2021
4481 00001C05 FEC2                <1>       inc   dl ; 3CFh
4482 00001C07 EE                  <1>       out   dx, al
4483 00001C08 FEC4                <1>       inc   ah
4484 00001C0A 80FC09              <1>       cmp   ah, 9 ; number of grdc regs
4485 00001C0D 72F0                <1>       jb    short _sm_10
4486                              <1>       ;
4487                              <1>       ; Disable CRTC write protection
4488                              <1>       ;mov  dx, 3D4h  ; VGAREG_VGA_CRTC_ADDRESS
4489                              <1>       ; 17/04/2021
4490 00001C0F B2D4                <1>       mov   dl, 0D4h
4491                              <1>       ;mov  al, 11h
4492                              <1>       ;our  dx, al
4493                              <1>       ;inc  dx
4494                              <1>       ;sub  al, al
4495                              <1>       ;out  dx, al
4496 00001C11 66B81100            <1>       mov   ax, 11h
4497 00001C15 66EF                <1>       out   dx, ax
4498 00001C17 89DE                <1>       mov   esi, ebx ; addr of params tbl for selected mode
4499 00001C19 83C60A              <1>       add   esi, 10 ; crtc regs
4500                              <1>       ; ah = 0
4501                              <1> _sm_11:
4502 00001C1C 88E0                <1>       mov   al, ah
4503                              <1>       ; dx = 3D4h = VGAREG_VGA_CRTC_ADDRESS
4504 00001C1E EE                  <1>       out   dx, al ; index
4505 00001C1F AC                  <1>       lodsb
4506                              <1>       ;inc  dx  ; VGAREG_VGA_CRTC_ADDRESS + 1
4507                              <1>       ; 17/04/2021
4508 00001C20 FEC2                <1>       inc   dl
4509 00001C22 EE                  <1>       out   dx, al ; value
4510 00001C23 80FC18              <1>       cmp   ah, 24 ; number of crtc registers - 1
4511 00001C26 7306                <1>       jnb   short _sm_12
4512 00001C28 FEC4                <1>       inc   ah
4513                              <1>       ;dec  dx ; 3D4h
4514                              <1>       ; 17/04/2021
4515 00001C2A FECA                <1>       dec   dl
4516 00001C2C EBEE                <1>       jmp   short _sm_11
4517                              <1> _sm_12:
4518                              <1>       ; Set the misc register
4519                              <1>       ;mov  dx, 3CCh ; VGAREG_READ_MISC_OUTPUT
4520                              <1>       ; 17/04/2021
4521 00001C2E B2CC                <1>       mov   dl, 0CCh
4522 00001C30 8A4309              <1>       mov   al, [ebx+9] ; misc reg
4523 00001C33 EE                  <1>       out   dx, al
4524                              <1>       ;
4525                              <1>       ; Enable video
4526                              <1>       ;mov  dx, 3C0h ; VGAREG_ACTL_ADDRESS
4527                              <1>       ; 17/04/2021
4528 00001C34 B2C0                <1>       mov   dl, 0C0h
4529 00001C36 B020                <1>       mov   al, 20h
4530 00001C38 EE                  <1>         out     dx, al   ; set bit 5 to 1
4531                              <1>       ;mov  dx, 3DAh ; VGAREG_ACTL_RESET
4532                              <1>       ; 17/04/2021
4533 00001C39 B2DA                <1>       mov   dl, 0DAh
4534 00001C3B EC                  <1>       in    al, dx
4535                              <1>       ;
4536                              <1>       ; 17/11/2020
4537                              <1>       ;cmp  byte [noclearmem], 0
4538                              <1>         ;ja  short _sm_15
4539                              <1>
4540 00001C3C F605[3F8D0100]80    <1>       test  byte [noclearmem], 80h
4541 00001C43 753C                <1>       jnz   short _sm_15
4542                              <1>
4543                              <1>       ; 29/07/2016
4544 00001C45 31C0                <1>       xor   eax, eax
4545 00001C47 B900400000          <1>       mov   ecx, 4000h ; 16K words (32K)
4546 00001C4C 803D[568D0100]02    <1>       cmp    byte [VGA_MTYPE], 2  ; CTEXT, MTEXT, CGA
4547 00001C53 7715                <1>       ja    short _sm_14    ; no ? (0A0000h)
4548 00001C55 BF00800B00          <1>       mov   edi, 0B8000h
4549 00001C5A 7409                <1>       je    short _sm_13 ; CGA graphics mode
4550                              <1>       ; 08/08/2016
4551 00001C5C A3[528D0100]        <1>       mov   [VGA_INT43H], eax ; 0 ; default font
4552 00001C61 66B82007            <1>       mov   ax, 0720h    ; CGA text mode
4553                              <1> _sm_13:
4554 00001C65 F366AB              <1>       rep   stosw
4555 00001C68 EB17                <1>       jmp   short _sm_15
4556                              <1>
4557                              <1> _sm_14:
4558 00001C6A BF00000A00          <1>       mov   edi, 0A0000h
4559                              <1>       ; ecx = 16384 dwords (64K)
4560                              <1>       ;mov  dx, 3C4h ; VGAREG_SEQU_ADDRESS
4561                              <1>       ; 17/04/2021
4562 00001C6F B2C4                <1>       mov   dl, 0C4h
4563 00001C71 B002                <1>       mov   al, 2
4564 00001C73 EE                  <1>       out   dx, al
4565                              <1>       ;mov  dx, 3C5h ; VGAREG_SEQU_DATA
4566                              <1>       ;inc  dx
4567                              <1>       ; 17/04/2021
4568 00001C74 FEC2                <1>       inc   dl ; 3C5h
4569 00001C76 EC                  <1>       in    al, dx ; mmask
4570                              <1>       ;push  ax
4571                              <1>       ; 12/04/2021
4572 00001C77 50                  <1>       push  eax
4573 00001C78 B00F                <1>       mov   al, 0Fh ; all planes
4574 00001C7A EE                  <1>       out   dx, al
```

```
4575 00001C7B 30C0                <1>       xor    al, al ; 0
4576 00001C7D F3AB                <1>       rep    stosd  ; ecx = 163684 (64K)
4577                              <1>       ;pop   ax
4578                              <1>       ; 12/04/2021
4579 00001C7F 58                  <1>       pop    eax
4580 00001C80 EE                  <1>       out    dx, al  ; mmask
4581                              <1> _sm_15:
4582                              <1>       ; ebx = addr of params tbl for selected mode
4583                              <1>       ; 10/08/2016
4584 00001C81 668B03              <1>       mov    ax, [ebx] ; num of columns, 'twidth'
4585 00001C84 A2[BC6A0000]        <1>       mov    [CRT_COLS], al
4586                              <1>       ;; 26/07/2016
4587                              <1>       ;; CRTC_ADDRESS = 3D4h (always)
4588                              <1>       ;mov   ah, [ebx+1] ; num of rows, 'theightm1'
4589 00001C89 FEC4                <1>       inc    ah ; 09/07/2016
4590 00001C8B 8825[C26A0000]      <1>       mov    [VGA_ROWS], ah
4591                              <1>       ; 10/08/2016
4592 00001C91 8A4302              <1>       mov    al, [ebx+2]
4593 00001C94 A2[BE6A0000]        <1>       mov    [CHAR_HEIGHT], al
4594                              <1>       ; 29/07/2016
4595                              <1>       ; length of regen buffer in bytes
4596 00001C99 668B4B03            <1>       mov    cx, [ebx+3] ; 'slength_l'
4597 00001C9D 66890D[408D0100]    <1>       mov    [CRT_LEN], cx
4598                              <1>       ;
4599                              <1>       ; 27/07/2016
4600 00001CA4 30E4                <1>       xor    ah, ah
4601 00001CA6 A0[D6800100]        <1>       mov    al, [ACTIVE_PAGE] ; may be > 0 for mode 3
4602                              <1>       ;mul   word [CRT_LEN] ; 4096 for mode 3
4603 00001CAB 66F7E1              <1>       mul    cx ; 29/07/2016
4604 00001CAE 66A3[C4800100]      <1>       mov    [CRT_START], ax
4605                              <1>       ;
4606 00001CB4 B060                <1>       mov    al, 60h
4607                              <1>       ;cmp   byte [noclearmem], 0
4608                              <1>       ;jna   short _sm_16
4609                              <1>       ;add   al, 80h
4610 00001CB6 0A05[3F8D0100]      <1>       or     al, [noclearmem] ; 17/11/2020
4611                              <1> _sm_16:
4612 00001CBC A2[BF6A0000]        <1>       mov    [VGA_VIDEO_CTL], al
4613 00001CC1 C605[C06A0000]F9    <1>       mov    byte [VGA_SWITCHES], 0F9h
4614 00001CC8 8025[C16A0000]7F    <1>       and    byte [VGA_MODESET_CTL], 7Fh
4615                              <1>
4616 00001CCF 5E                  <1>       pop    esi ; *
4617                              <1>
4618                              <1>       ; 26/07/2016
4619                              <1>       ; 07/07/2016
4620 00001CD0 668B0D[D36A0000]    <1>       mov    cx, [CURSOR_MODE] ; restore cursor mode (initial value)
4621 00001CD7 66870E              <1>       xchg   cx, [esi] ; cl = start line, ch = end line
4622                              <1>                        ; reset to initial value
4623 00001CDA 86E9                <1>       xchg   ch, cl  ; ch = start line, cl = end line
4624 00001CDC 66890D[D36A0000]    <1>       mov    [CURSOR_MODE], cx ; save (fixed) cursor mode
4625                              <1>
4626                              <1>       ; 27/07/2016
4627 00001CE3 803D[568D0100]02    <1>       cmp    byte [VGA_MTYPE], 2 ; CTEXT, MTEXT
4628 00001CEA 7317                <1>       jnb    short _sm_17
4629                              <1>
4630                              <1>       ; Set cursor shape
4631                              <1>       ;mov   cx, 0607h
4632                              <1>       ;call  _set_ctype
4633                              <1>
4634                              <1>       ; 29/07/2016
4635 00001CEC B40A                <1>       mov    ah, 10 ; 6845 register for cursor set
4636 00001CEE E832060000          <1>       call   m16    ; output cx register
4637                              <1>
4638                              <1>       ; 25/07/2016
4639 00001CF3 803D[BA6A0000]03    <1>         cmp     byte [CRT_MODE], 03h
4640 00001CFA 7507                <1>       jne    short _sm_17
4641                              <1>       ; 26/07/2016
4642                              <1>
4643 00001CFC A0[D6800100]        <1>       mov    al, [ACTIVE_PAGE]
4644 00001D01 EB0B                <1>       jmp    short _sm_18
4645                              <1> _sm_17:
4646                              <1>       ; Set cursor pos for page 0..7
4647                              <1>       ;sub   ax, ax ; eax = 0
4648 00001D03 29C0                <1>       sub    eax, eax ; 17/11/2020
4649 00001D05 BF[C6800100]        <1>       mov    edi, CURSOR_POSN
4650 00001D0A AB                  <1>       stosd
4651 00001D0B AB                  <1>       stosd
4652 00001D0C AB                  <1>       stosd
4653 00001D0D AB                  <1>       stosd
4654                              <1>       ;; Set active page 0
4655                              <1>       ;mov   [ACTIVE_PAGE], al ; 0
4656                              <1> _sm_18:
4657                              <1>       ; 29/07/2016
4658 00001D0E 803D[568D0100]02    <1>       cmp    byte [VGA_MTYPE], 2 ; CTEXT, MTEXT
4659 00001D15 737F                <1>         jnb  _sm_23
4660                              <1>
4661                              <1>       ;cmp   byte [CHAR_HEIGHT], 16
4662                              <1>       ;je    short _sm_19
4663                              <1>
4664                              <1>       ;; copy and activate 8x16 font
4665                              <1>
4666                              <1>       ; 26/07/2016
4667 00001D17 B004                <1>       mov    al, 04h
4668                              <1>       ;sub   bl, bl
4669                              <1>       ; AX = 1104H ; Load ROM 8x16 Character Set
4670                              <1>       ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
4671 00001D19 E8B4160000          <1>       call   load_text_8_16_pat
4672                              <1>
4673                              <1>       ; video_func_1103h:
4674                              <1>       ; biosfn_set_text_block_specifier:
4675                              <1>       ; BL = font block selector code
4676                              <1>       ; NOTE: TRDOS 386 only uses and sets font block 0
4677                              <1>       ; (It is as BL = 0 for TRDOS 386)
4678 00001D1E 66BAC403            <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
4679                              <1>       ;;mov  ah, bl
```

```
4680                                    <1>       ;sub   ah, ah ; 0
4681                                    <1>       ;mov   al, 03h
4682                                    <1>       ; 19/11/2020
4683 00001D22 66B80300                  <1>       mov    ax, 03h
4684 00001D26 66EF                      <1>       out    dx, ax
4685                                    <1> _sm_19:
4686                                    <1>       ; 29/07/2016
4687                                    <1>       ; 26/07/2016
4688                                    <1>       ; 24/06/2016
4689                                    <1>       ;mov   edi, 0B8000h
4690                                    <1>       ;mov   cx, 4000h ; 16K words (32K)
4691                                    <1>       ;
4692 00001D28 30C0                     <1>       xor    al, al
4693 00001D2A 3805[3D8D0100]           <1>         cmp     [p_crt_mode], al ; 0
4694 00001D30 7705                     <1>         ja      short _sm_20 ; 03h, 80h or 83h
4695                                    <1>
4696                                    <1>       ; case 1 - 19/11/2020
4697                                    <1>       ;
4698                                    <1>       ; If [pc_crt_mode] = 0, that means, previous mode is 03h
4699                                    <1>       ; and current mode is not 03h (case 1)
4700                                    <1>
4701                                    <1>       ; 30/07/2016
4702                                    <1>       ; 24/06/2016
4703                                    <1>       ; TRDOS 386 (TRDOS v2) 'set mode' modification
4704                                    <1>       ; (for multiscreen feature):
4705                                    <1>       ; If '_set_mode' procedure is called for video mode 3
4706                                    <1>       ;     while video mode is 3, video page will be cleared
4707                                    <1>       ;     and cursor position of video page will be reset.
4708                                    <1>       ; If '_set_mode' procedure is called for a video mode
4709                                    <1>       ;     except video mode 3, while current video mode
4710                                    <1>       ;     is 3, all video pages of mode 3 will be copied
4711                                    <1>       ;     to 98000h address as backup, before mode change.
4712                                    <1>       ; If '_set_mode' procedure is called for video mode 3
4713                                    <1>       ;     while video mode is not 3 and if there is video
4714                                    <1>       ;     page backup for video mode 3, all (of 8) mode 3
4715                                    <1>       ;     video pages will be restored from 98000h.
4716                                    <1>
4717                                    <1>       ; 19/11/2020
4718                                    <1>       ;mov   [ACTIVE_PAGE], al ; 0
4719                                    <1>
4720                                    <1>       ; Here,
4721                                    <1>       ; video memory already cleared if [noclearmem] <> 80h
4722                                    <1>
4723                                    <1>       ;mov   ax, 0720h
4724                                    <1>       ;mov   cx, 4000h ; 16K words (32K)
4725                                    <1>       ;mov   edi, 0B8000h
4726                                    <1>       ;rep   stosw
4727                                    <1>       ;sub   al, al
4728                                    <1>
4729                                    <1>       ;jmp   short _sm_23
4730                                    <1>
4731                                    <1>       ; Set hardware side for the new active video page
4732                                    <1>
4733 00001D32 E9F9010000               <1>       jmp    _set_active_page ; 19/11/2020
4734                                    <1>
4735                                    <1> _sm_20:
4736                                    <1>       ; 19/11/2020
4737                                    <1>       ; case 2 or case 3 or case 4 - 19/11/2020
4738                                    <1>
4739                                    <1>       ; 19/11/2020
4740 00001D37 803D[BA6A0000]03         <1>       cmp    byte [CRT_MODE], 3  ; new video mode
4741 00001D3E 754F                     <1>       jne    short _sm_22 ; al = 0 (& video mode <> 03h)
4742                                    <1>                         ; case 4 - 19/11/2020
4743                                    <1>                         ; ([p_crt_mode] = 83h)
4744                                    <1>
4745                                    <1>       ; case 2 or case 3 - 19/11/2020
4746                                    <1>
4747                                    <1>       ;movzx ebx, byte [ACTIVE_PAGE]
4748                                    <1>       ; 19/11/2020
4749 00001D40 A0[3E8D0100]             <1>       mov    al, [p_crt_page] ; previous mode 3 active page
4750                                    <1>       ;movzx ebx, al
4751                                    <1>       ;shl   bl, 1 ; * 2
4752                                    <1>       ;add   ebx, CURSOR_POSN
4753                                    <1>
4754                                    <1>       ; 29/07/2016
4755 00001D45 F605[3D8D0100]7F         <1>       test    byte [p_crt_mode], 7Fh ; 83h or 80h or 03h
4756 00001D4C 740F                     <1>       jz     short _sm_21 ; do not restore video pages
4757                                    <1>                           ; case 2 - 19/11/2020
4758                                    <1>       ; case 3 - 19/11/2020
4759                                    <1>
4760                                    <1>       ; ([p_crt_mode] = 03h)
4761                                    <1>
4762                                    <1>       ; New video mode is 3 while current video mode is not 3
4763                                    <1>       ; (multi screen) video pages will be restored from 098000h
4764                                    <1>
4765                                    <1>       ; 19/11/2020
4766 00001D4E A2[D6800100]             <1>       mov    [ACTIVE_PAGE], al ; current mode 3 active page
4767                                    <1>
4768                                    <1>       ; 12/12/2020
4769                                    <1>       ;; restore video pages
4770                                    <1>       ;mov   esi, 98000h ; 30/07/2016
4771                                    <1>       ;mov   edi, 0B8000h
4772                                    <1>       ;mov   cx, 2000h ; 8K dwords (32K)
4773                                    <1>       ;rep   movsd
4774                                    <1>       ;
4775                                    <1>       ;; 19/11/2020
4776                                    <1>       ;mov   [p_crt_mode], cl ; reset ('case 3' end condition)
4777                                    <1>       ;
4778                                    <1>       ;; restore cursor positions
4779                                    <1>       ;mov   esi, cursor_pposn
4780                                    <1>       ;mov   edi, CURSOR_POSN
4781                                    <1>       ;;mov  ecx, 4 ; restore all cursor positions (16 bytes)
4782                                    <1>       ;mov   cl, 4
4783                                    <1>       ;rep   movsd
4784                                    <1>
```

```
4785                                    <1>       ; 12/12/2020
4786 00001D53 E89C000000                <1>       call   _restore_mode3_multiscreen
4787                                    <1>
4788                                    <1>       ;jmp   short _sm_23 ; do not clear current video pages
4789                                    <1>
4790                                    <1>       ; 19/11/2020
4791 00001D58 E9D3010000                <1>       jmp    _set_active_page
4792                                    <1>
4793                                    <1> _sm_21:
4794                                    <1>       ; 19/11/2020
4795                                    <1>       ; case 2
4796                                    <1>       ;
4797                                    <1>       ; ([p_crt_mode] = 80h)
4798                                    <1>       ;
4799                                    <1>       ; User has requested to set video mode 3 again while
4800                                    <1>       ; current video mode is 3.. that means, set mode 03h
4801                                    <1>       ; parameters again and clear video page.
4802                                    <1>       ; ('noclearmem' option effects the result)
4803                                    <1>
4804                                    <1>       ; 19/11/2020
4805 00001D5D F605[3F8D0100]80          <1>       test   byte [noclearmem], 80h
4806 00001D64 7529                      <1>       jnz    short _sm_22 ; 'do not clear video memory'
4807                                    <1>                           ; continue with current text
4808                                    <1>                           ; (user's option/choice)
4809                                    <1>       ; clear video page
4810                                    <1>       ;mov   cx, [CRT_LEN] ; 4096
4811                                    <1>       ;shr   cx, 1 ; 2048 ; 16/11/2020
4812 00001D66 66B82007                  <1>       mov    ax, 0720h
4813                                    <1>       ; 26/11/2020
4814 00001D6A B900080000                <1>       mov    ecx, 2048 ; 4096/2
4815 00001D6F BF00800B00                <1>       mov    edi, 0B8000h ; [crt_base]
4816 00001D74 66033D[C4800100]          <1>       add    di, [CRT_START]
4817 00001D7B F366AB                    <1>       rep    stosw  ; FILL THE REGEN BUFFER WITH BLANKS
4818                                    <1>       ;
4819                                    <1>       ; 19/11/2020
4820 00001D7E A0[D6800100]              <1>       mov    al, [ACTIVE_PAGE] ; 0 to 7 (for video mode 3)
4821 00001D83 0FB6D8                    <1>       movzx  ebx, al
4822 00001D86 D0E3                      <1>       shl    bl, 1
4823 00001D88 66898B[C6800100]          <1>       mov    [ebx+CURSOR_POSN], cx ; reset cursor position
4824                                    <1> _sm_22:
4825                                    <1>       ;mov   [p_crt_mode], al ; 0 ; reset
4826                                    <1>       ; 19/11/2020
4827                                    <1>       ;and   byte [p_crt_mode], 3 ; 83h -> 3, 80h -> 0
4828 00001D8F 8025[3D8D0100]7F         <1>       and    byte [p_crt_mode], 7Fh ; 83h -> 3, 80h -> 0
4829                                    <1> _sm_23:
4830                                    <1>       ; al = video page number
4831                                    <1>       ; [CRT_LEN] = length of regen buffer in bytes
4832                                    <1>       ;call  _set_active_page
4833                                    <1>       ; 16/11/2020
4834 00001D96 E995010000                <1>       jmp    _set_active_page
4835                                    <1>
4836                                    <1> ;-----       NORMAL RETURN FROM ALL VIDEO RETURNS
4837                                    <1>       ;retn
4838                                    <1>
4839                                    <1> save_mode3_multiscreen:
4840                                    <1>       ; 12/12/2020 (TRDOS v2.0.3)
4841                                    <1>       ; save mode 03h video pages and cursor positions
4842                                    <1>       ;
4843                                    <1>       ; Modified registers: ecx (=0), esi, edi
4844                                    <1>
4845                                    <1>       ; 12/12/2020
4846                                    <1>       ; moved here from '_set_mode'
4847                                    <1>       ; 03/07/2016
4848                                    <1>       ; save video pages
4849 00001D9B BE00800B00                <1>       mov    esi, 0B8000h
4850 00001DA0 BF00800900                <1>       mov    edi, 98000h ; 30/07/2016
4851 00001DA5 B900200000                <1>       mov    ecx, (0B8000h-0B0000h)/4
4852 00001DAA F3A5                      <1>       rep    movsd
4853                                    <1>
4854 00001DAC C605[3D8D0100]03          <1>       mov    byte [p_crt_mode], 3 ; previous mode, backup sign
4855                                    <1>       ; 26/11/2020
4856 00001DB3 860D[D6800100]            <1>       xchg   cl, [ACTIVE_PAGE]
4857 00001DB9 880D[3E8D0100]            <1>       mov    [p_crt_page], cl  ; save as previous active page
4858                                    <1>
4859                                    <1>       ; save cursor positions
4860 00001DBF BE[C6800100]              <1>       mov    esi, CURSOR_POSN
4861 00001DC4 BF[428D0100]              <1>       mov    edi, cursor_pposn ; cursor positions backup
4862 00001DC9 B104                      <1>       mov    cl, 4
4863 00001DCB F3A5                      <1>       rep    movsd
4864 00001DCD C3                        <1>       retn
4865                                    <1>
4866                                    <1> restore_mode3_multiscreen:
4867                                    <1>       ; 12/12/2020 (TRDOS v2.0.3)
4868                                    <1>       ; restore mode 03h video pages and cursor positions
4869                                    <1>       ;
4870                                    <1>       ; Input:
4871                                    <1>       ;    settings from the last 'save_mode3_multiscreen'
4872                                    <1>       ;
4873                                    <1>       ; Output:
4874                                    <1>       ;    AL = active video page = [ACTIVE_PAGE]
4875                                    <1>       ;
4876                                    <1>       ; Modified registers: al, ecx (=0), esi, edi
4877                                    <1>
4878 00001DCE A0[3E8D0100]              <1>       mov    al, [p_crt_page] ; previous mode 3 active page
4879 00001DD3 A2[D6800100]              <1>       mov    [ACTIVE_PAGE], al ; current mode 3 active page
4880                                    <1>
4881                                    <1>       ; 12/12/2020
4882                                    <1>       ; moved here from 'vesa_vbe3_pmi'
4883                                    <1>
4884                                    <1>       ; 07/12/2020
4885                                    <1>       ; restore CRT_START according to ACTIVE_PAGE
4886                                    <1>       ;mov   [CRT_START], cx ; 0
4887                                    <1>       ; 12/12/2020
4888 00001DD8 66C705[C4800100]00-       <1>       mov    word [CRT_START], 0
4888 00001DE0 00                        <1>
```

```
                                <1>
                                <1>       ; check active page and set it again if it is not 0
00001DE1 08C0                   <1>       or     al, al
                                <1>       ;;jz   short vbe3_pmi_7
                                <1>       ;jz    short _restore_mode3_multiscreen
00001DE3 740F                   <1>       jz     short r_m3_ms_1
00001DE5 88C1                   <1>       mov    cl, al
                                <1> ;vbe3_pmi_5:
                                <1> r_m3_ms_0:
00001DE7 668105[C4800100]00-    <1>       add    word [CRT_START], 4096
00001DEF 10                     <1>
00001DF0 FEC9                   <1>       dec    cl
                                <1>       ;jnz   short vbe3_pmi_5
00001DF2 75F3                   <1>       jnz    short r_m3_ms_0
                                <1> r_m3_ms_1:
                                <1>       ; 12/12/2020
                                <1>       ; moved here from '_set_mode'
                                <1> _restore_mode3_multiscreen:
                                <1>       ; Modified registers: ecx, esi, edi
                                <1>
                                <1>       ; restore video pages
00001DF4 BE00800900             <1>       mov    esi, 98000h ; 30/07/2016
00001DF9 BF00800B00             <1>       mov    edi, 0B8000h
                                <1>       ;mov   cx, 2000h ; 8K dwords (32K)
00001DFE B900200000             <1>       mov    ecx, 2000h
00001E03 F3A5                   <1>       rep    movsd
                                <1>
                                <1>       ; 19/11/2020
00001E05 880D[3D8D0100]         <1>       mov    [p_crt_mode], cl ; reset ('case 3' end condition)
                                <1>
                                <1>       ; restore cursor positions
00001E0B BE[428D0100]           <1>       mov    esi, cursor_pposn
00001E10 BF[C6800100]           <1>       mov    edi, CURSOR_POSN
                                <1>       ;mov   ecx, 4 ; restore all cursor positions (16 bytes)
00001E15 B104                   <1>       mov    cl, 4
00001E17 F3A5                   <1>       rep    movsd
00001E19 C3                     <1>       retn
                                <1>
                                <1> cursor_shape_fix:
                                <1>       ; 12/04/2021
                                <1>       ; 07/07/2016
                                <1>       ; (Cursor start and cursor end line values -6,7-
                                <1>       ; will be fixed depending on character height)
                                <1>       ;
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
                                <1>       ; 'vgabios.c', ' biosfn_set_cursor_shape (CH,CL)'
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;      AL = cursor start line (=6)
                                <1>       ;      AH = cursor end line (=7)
                                <1>       ; OUTPUT ->
                                <1>       ;      AL = cursor start line (=14)
                                <1>       ;      AH = cursor end line (=15)
                                <1>       ;
                                <1>       ;; if((modeset_ctl&0x01)&&(cheight>8)&&(CL<8)&&(CH<0x20))
                                <1>
                                <1>       ;test  byte [VGA_MODESET_CTL], 1 ; VGA active
                                <1>       ;jz    short csf_3
00001E1A 803D[BE6A0000]08       <1>       cmp    byte [CHAR_HEIGHT], 8
00001E21 7647                   <1>       jna    short csf_3
00001E23 80FC08                 <1>       cmp    ah, 8
00001E26 7342                   <1>       jnb    short csf_3
00001E28 3C20                   <1>       cmp    al, 20h
00001E2A 733E                   <1>       jnb    short csf_3
                                <1>       ;
                                <1>       ;push  ax
                                <1>       ; 12/04/2021
00001E2C 50                     <1>       push   eax
                                <1>       ; {
                                <1>       ; if(CL!=(CH+1))
00001E2D FEC0                   <1>       inc    al
00001E2F 38C4                   <1>       cmp    ah, al   ; ah != al + 1
00001E31 740F                   <1>         je     short csf_1
                                <1>       ; CH = ((CH+1) * cheight / 8) -1;
00001E33 8A25[BE6A0000]         <1>       mov    ah, [CHAR_HEIGHT]
00001E39 F6E4                   <1>       mul    ah
00001E3B C0E803                 <1>       shr    al, 3 ; / 8
00001E3E FEC8                   <1>       dec    al ; - 1
00001E40 EB0E                   <1>       jmp    short csf_2
                                <1> csf_1:
                                <1>       ; }
                                <1>       ; else       ; ah = al + 1
                                <1>       ; {
00001E42 FEC4                   <1>       inc    ah     ; ah = ah + 1
                                <1>       ; CH = ((CL+1) * cheight / 8) - 2;
00001E44 A0[BE6A0000]           <1>       mov    al, [CHAR_HEIGHT]
00001E49 F6E4                   <1>       mul    ah
00001E4B C0E803                 <1>       shr    al, 3 ; / 8
00001E4E 2C02                   <1>       sub    al, 2 ; - 2
                                <1>       ; al = 14 (if [CHAR_HEIGHT] = 16)
                                <1> csf_2:
00001E50 880424                 <1>       mov    [esp], al
00001E53 8A642401               <1>       mov    ah, [esp+1]
                                <1>       ; CL = ((CL+1) * cheight / 8) - 1;
00001E57 FEC4                   <1>       inc    ah
00001E59 A0[BE6A0000]           <1>       mov    al, [CHAR_HEIGHT]
00001E5E F6E4                   <1>       mul    ah
00001E60 C0E803                 <1>       shr    al, 3 ; / 8
00001E63 FEC8                   <1>       dec    al ; - 1
00001E65 88442401               <1>       mov    [esp+1], al
                                <1>       ; ah = 15 (if [CHAR_HEIGHT] = 16)
                                <1>       ;
                                <1>       ;pop   ax
```

```
                                <1>       ; 12/04/2021
00001E69 58                     <1>       pop    eax
                                <1> csf_3:
00001E6A C3                     <1>       retn
                                <1>
                                <1> SET_CTYPE:
                                <1>       ; 12/09/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
00001E6B 803D[BA6A0000]07       <1>       cmp    byte [CRT_MODE], 7
00001E72 0F8771FCFFFF           <1>       ja     VIDEO_RETURN ; 12/09/2016
00001E78 E805000000             <1>       call   _set_ctype
00001E7D E967FCFFFF             <1>         jmp     VIDEO_RETURN
                                <1>
                                <1> _set_ctype:
                                <1>       ; 02/09/2014 (Retro UNIX 386 v1)
                                <1>       ;
                                <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                                <1>
                                <1>       ; (CH) = BITS 4-0 = START LINE FOR CURSOR
                                <1>       ;  ** HARDWARE WILL ALWAYS CAUSE BLINK
                                <1>       ;  ** SETTING BIT 5 OR 6 WILL CAUSE ERRATIC BLINKING
                                <1>       ;     OR NO CURSOR AT ALL
                                <1>       ; (CL) = BITS 4-0 = END LINE FOR CURSOR
                                <1>
                                <1> ;--------------------------------------------
                                <1> ; SET_CTYPE
                                <1> ;     THIS ROUTINE SETS THE CURSOR VALUE
                                <1> ; INPUT
                                <1> ;     (CX) HAS CURSOR VALUE CH-START LINE, CL-STOP LINE
                                <1> ; OUTPUT
                                <1> ;     NONE
                                <1> ;--------------------------------------------
                                <1>
                                <1>       ; 07/07/2016
                                <1>       ; Fixing cursor start and stop line depending on
                                <1>       ; current character height (=16)
                                <1>       ; (Note: Default/initial values are 6 and 7.
                                <1>       ; If set values are 6 (start) & 7 (stop) and
                                <1>       ; [CHAR_HEIGHT] = 16 :
                                <1>       ; After fixing, start line will be 14, stop line
                                <1>       ; will be 15.)
00001E82 6689C8                 <1>       mov    ax, cx
00001E85 86C4                   <1>       xchg   al, ah
                                <1>       ; AL = start line, AH = stop line
00001E87 E88EFFFFFF             <1>       call   cursor_shape_fix
                                <1>       ; AL = start line (fixed), AH = stop line (fixed)
00001E8C 6689C1                 <1>       mov    cx, ax
00001E8F 86E9                   <1>       xchg   ch, cl
                                <1>       ; CH = start line (fixed), CL = stop line (fixed)
                                <1>       ;
00001E91 B40A                   <1>       mov    ah, 10 ; 6845 register for cursor set
00001E93 66890D[D36A0000]       <1>       mov    [CURSOR_MODE], cx ; save in data area
                                <1>       ;call  m16    ; output cx register
                                <1>       ;retn
00001E9A E986040000             <1>         jmp     m16
                                <1>
                                <1> SET_CPOS:
                                <1>       ; 12/09/2016
                                <1>       ; 07/07/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
00001E9F 80FF07                 <1>       cmp    bh, 7 ; video page > 7 ; 07/07/2016
00001EA2 0F8741FCFFFF           <1>       ja     VIDEO_RETURN
                                <1>       ;
00001EA8 803D[BA6A0000]07       <1>       cmp    byte [CRT_MODE], 7
00001EAF 770A                   <1>       ja     short vga_set_cpos ; 12/09/2016
00001EB1 E844040000             <1>       call   _set_cpos
00001EB6 E92EFCFFFF             <1>         jmp     VIDEO_RETURN
                                <1>
                                <1> vga_set_cpos:
                                <1>       ; 12/09/2016
                                <1>       ; 09/07/2016
                                <1>       ; set cursor position
                                <1>       ; NOTE: Hardware cursor position will not be set
                                <1>       ;   in any VGA modes (>7)
                                <1>       ;   But, cursor position will be saved into
                                <1>       ;   [CURSOR_POSN].
                                <1>       ;   TRDOS 386 (TRDOS v2.0) uses only one page
                                <1>       ;   (page 0) for all graphics modes.
                                <1>
00001EBB 668915[C6800100]       <1>       mov    [CURSOR_POSN], dx ; save cursor pos for pg 0
                                <1>       ; 04/08/2016
                                <1>       ;mov   bh, [ACTIVE_PAGE] ; = 0
                                <1>       ;call  _set_cpos
00001EC2 E922FCFFFF             <1>       jmp     VIDEO_RETURN
                                <1>
                                <1> READ_CURSOR:
                                <1>       ; 12/09/2016
                                <1>       ; 07/07/2016
                                <1>       ; 12/05/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ;
                                <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                                <1>
                                <1> ;--------------------------------------------------------
                                <1> ; READ_CURSOR
                                <1> ;     THIS ROUTINE READS THE CURRENT CURSOR VALUE FROM THE
                                <1> ;     845, FORMATS IT, AND SENDS IT BACK TO THE CALLER
                                <1> ; INPUT
                                <1> ;     BH - PAGE OF CURSOR
                                <1> ; OUTPUT
                                <1> ;     DX - ROW, COLUMN OF THE CURRENT CURSOR POSITION
                                <1> ;     CX - CURRENT CURSOR MODE
                                <1> ;--------------------------------------------------------
                                <1>
                                <1>       ; BH = Video page number (0 to 7)
```

```
                                <1>
                                <1>       ; 07/07/2016
00001EC7 80FF07                 <1>       cmp    bh, 7 ; video page > 7 (invalid)
00001ECA 7606                   <1>       jna    short read_cursor_1
                                <1>       ; invalid video page (input)
00001ECC 31C9                   <1>       xor    ecx, ecx ; 0
00001ECE 31D2                   <1>       xor    edx, edx ; 0
00001ED0 EB15                   <1>       jmp    short read_cursor_2
                                <1> read_cursor_1:
                                <1>       ; 12/09/2016
00001ED2 803D[BA6A0000]07       <1>       cmp    byte [CRT_MODE], 7 ; vga mode
00001ED9 7727                   <1>       ja     short vga_get_cpos
                                <1>       ;
00001EDB E815000000             <1>       call   get_cpos
00001EE0 0FB70D[D36A0000]       <1>       movzx  ecx, word [CURSOR_MODE]
                                <1> read_cursor_2:
00001EE7 5D                     <1>       pop    ebp
00001EE8 5F                     <1>       pop    edi
00001EE9 5E                     <1>       pop    esi
00001EEA 5B                     <1>       pop    ebx
00001EEB 58                     <1>       pop    eax  ; DISCARD SAVED CX AND DX
00001EEC 58                     <1>       pop    eax
00001EED A1[308D0100]           <1>       mov    eax, [video_eax] ; 12/05/2016
                                <1>       ;;15/01/2017
                                <1>       ;;mov  byte [intflg], 0 ; 07/01/2017
00001EF2 1F                     <1>       pop    ds
00001EF3 07                     <1>       pop    es
00001EF4 CF                     <1>       iretd
                                <1>
                                <1> get_cpos:
                                <1>       ; 12/05/2016
                                <1>       ; 16/01/2016
                                <1>       ; BH = Video page number (0 to 7)
                                <1>       ;
00001EF5 D0E7                   <1>       shl    bh, 1 ; WORD OFFSET
00001EF7 0FB6F7                 <1>       movzx  esi, bh
00001EFA 0FB796[C6800100]       <1>       movzx  edx, word [esi+CURSOR_POSN]
00001F01 C3                     <1>       retn
                                <1>
                                <1> vga_get_cpos:
                                <1>       ; 12/09/2016
                                <1>       ; get cursor position (vga)
00001F02 0FB715[C6800100]       <1>       movzx  edx, word [CURSOR_POSN] ; cursor pos for pg 0
00001F09 31C9                   <1>       xor    ecx, ecx ; Cursor Mode = 0 (invalid)
00001F0B EBDA                   <1>       jmp     short read_cursor_2
                                <1>
                                <1> ACT_DISP_PAGE:
                                <1>       ; 07/07/2016
                                <1>       ; 26/06/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ;
                                <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                                <1>       ;
                                <1> ;--------------------------------------------------
                                <1> ; ACT_DISP_PAGE
                                <1> ;     THIS ROUTINE SETS THE ACTIVE DISPLAY PAGE, ALLOWING
                                <1> ;     THE FULL USE OF THE MEMORY SET ASIDE FOR THE VIDEO ATTACHMENT
                                <1> ; INPUT
                                <1> ;     AL HAS THE NEW ACTIVE DISPLAY PAGE
                                <1> ; OUTPUT
                                <1> ;     THE 6845 IS RESET TO DISPLAY THAT PAGE
                                <1> ;--------------------------------------------------
                                <1>       ; 07/07/2016
00001F0D 3C07                   <1>       cmp    al, 7  ; > 7 = invalid video page number
                                <1>       ;ja    VIDEO_RETURN
00001F0F 7715                   <1>         ja   short adp_2 ; 18/11/2020
                                <1>       ;cmp   byte [CRT_MODE], 3
                                <1>       ;je    short adp_1
                                <1>       ; 18/11/2020
00001F11 8A25[BA6A0000]         <1>       mov    ah, [CRT_MODE]
00001F17 80FC03                 <1>       cmp    ah, 3
00001F1A 7605                   <1>       jna    short adp_1 ; mode 01h, 00h (01h), 02h (03h), 03h
00001F1C 80FC07                 <1>       cmp    ah, 7      ; mode 07h (03h)
00001F1F 7505                   <1>       jne    short adp_2
                                <1>       ;and   al, al
                                <1>         ;jnz VIDEO_RETURN
                                <1>       ;;sub  al, al ; 0 ; force to page 0
                                <1> adp_1:
00001F21 E805000000             <1>       call   set_active_page
                                <1> adp_2:
00001F26 E9BEFBFFFF             <1>         jmp     VIDEO_RETURN
                                <1>
                                <1> set_active_page:   ; tty_sw
                                <1>       ; 09/12/2017
                                <1>       ; 26/07/2016
                                <1>       ; 26/06/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 30/06/2015
                                <1>       ; 04/03/2014  (act_disp_page --> tty_sw)
                                <1>       ; 10/12/2013
                                <1>       ; 04/12/2013
                                <1>       ;
00001F2B A2[D6800100]           <1>       mov    [ACTIVE_PAGE], al ; save active page value ; [ptty]
                                <1> _set_active_page:
                                <1>       ; 27/06/2015
00001F30 0FB6D8                 <1>       movzx  ebx, al
                                <1>       ;
                                <1>       ;cbw   ; 07/09/2014 (ah=0)
00001F33 28E4                   <1>       sub    ah, ah ; 09/12/2017
00001F35 66F725[408D0100]       <1>       mul    word [CRT_LEN]  ; get saved length of regen buffer
                                <1>                              ; display page times regen length
                                <1>       ; 10/12/2013
00001F3C 66A3[C4800100]         <1>       mov    [CRT_START], ax ; save start address for later
00001F42 6689C1                 <1>       mov    cx, ax ; start address to cx
                                <1> _M16:
```

```
                                <1>       ;sar   cx, 1
00001F45 66D1E9                 <1>       shr    cx, 1  ; divide by 2 for 6845 handling
00001F48 B40C                   <1>       mov    ah, 12 ; 6845 register for start address
00001F4A E8D6030000             <1>       call   m16
                                <1>       ;sal   bx, 1
                                <1>       ; 01/09/2014
00001F4F D0E3                   <1>       shl    bl, 1  ; *2 for word offset
00001F51 81C3[C6800100]         <1>       add    ebx, CURSOR_POSN
00001F57 668B13                 <1>       mov    dx, [ebx] ; get cursor for this page
                                <1>       ; 16/01/2016
                                <1>       ;call  m18
                                <1>       ;retn
00001F5A E9B2030000             <1>       jmp    m18
                                <1>
                                <1> position:
                                <1>       ; 17/04/2021 - TRDOS 386 v2.0.4
                                <1>       ; 24/06/2016
                                <1>       ; 12/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; 27/06/2015
                                <1>       ; 02/09/2014
                                <1>       ; 30/08/2014 (Retro UNIX 386 v1)
                                <1>       ; 04/12/2013 (Retro UNIX 8086 v1)
                                <1>       ;
                                <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                                <1>       ;
                                <1> ;-----------------------------------------
                                <1> ; POSITION
                                <1> ;     THIS SERVICE ROUTINE CALCULATES THE REGEN BUFFER ADDRESS
                                <1> ;     OF A CHARACTER IN THE ALPHA MODE
                                <1> ; INPUT
                                <1> ;     AX = ROW, COLUMN POSITION
                                <1> ; OUTPUT
                                <1> ;     AX = OFFSET OF CHAR POSITION IN REGEN BUFFER
                                <1> ;-----------------------------------------
                                <1>
                                <1>       ; DX = ROW, COLUMN POSITION
                                <1>       ;movzx eax, byte [CRT_COLS] ; 27/06/2015
00001F5F 31C0                   <1>       xor    eax, eax ; 02/09/2014
00001F61 B050                   <1>       mov    al, 80   ; determine bytes to row
00001F63 F6E6                   <1>       mul    dh      ; row value
                                <1>       ;xor   dh, dh   ; 0
                                <1>       ;add   ax, dx  ; add column value to the result
                                <1>       ; 16/04/2021
00001F65 00D0                   <1>       add    al, dl
00001F67 80D400                 <1>       adc    ah, 0
00001F6A 66D1E0                 <1>       shl    ax, 1  ; * 2 for attribute bytes
                                <1>              ; EAX = AX = OFFSET OF CHAR POSITION IN REGEN BUFFER
00001F6D C3                     <1>       retn
                                <1>
                                <1> find_position:
                                <1>       ; 17/04/2021 - TRDOS 386 v2.0.4
                                <1>       ; 24/06/2016
                                <1>       ; 12/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; 27/06/2015
                                <1>       ; 07/09/2014
                                <1>       ; 02/09/2014
                                <1>       ; 30/08/2014 (Retro UNIX 386 v1)
                                <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                                <1>
00001F6E 0FB6CF                 <1>       movzx  ecx, bh ; video page number
                                <1>       ; 17/04/2021
                                <1>       ;mov   esi, ecx
                                <1>       ;shl   si, 1
                                <1>       ;mov   dx, [esi+CURSOR_POSN]
                                <1>       ;jz    short p21
                                <1>       ;xor   si, si
                                <1>       ; 17/04/2021
00001F71 31F6                   <1>       xor    esi, esi
00001F73 D0E1                   <1>       shl    cl, 1
00001F75 668B91[C6800100]       <1>       mov    dx, [ecx+CURSOR_POSN]
00001F7C 740B                   <1>       jz     short p21
00001F7E D0E9                   <1>       shr    cl, 1
                                <1> p20:
00001F80 660335[408D0100]       <1>       add    si, [CRT_LEN] ; 24/06/2016
                                <1>       ;add   si, 80*25*2 ; add length of buffer for one page
00001F87 E2F7                   <1>       loop   p20
                                <1> p21:
00001F89 6621D2                 <1>       and    dx, dx
00001F8C 7407                   <1>       jz     short p22
00001F8E E8CCFFFFFF             <1>       call   position ; determine location in regen in page
00001F93 01C6                   <1>       add    esi, eax ; add location to start of regen page
                                <1> p22:
                                <1>       ;mov   dx, [addr_6845] ; get base address of active display
                                <1>       ;mov   dx, 03D4h ; I/O address of color card
                                <1>       ;add   dx, 6  ; point at status port
00001F95 66BADA03               <1>       mov    dx, 03DAh ; status port
                                <1>       ; cx = 0
00001F99 C3                     <1>       retn
                                <1>
                                <1> SCROLL_UP:
                                <1>       ; 07/07/2016
                                <1>       ; 26/06/2016
                                <1>       ; 12/05/2016
                                <1>       ; 30/01/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 07/09/2014
                                <1>       ; 02/09/2014
                                <1>       ; 01/09/2014 (Retro UNIX 386 v1 - beginning)
                                <1>       ; 04/04/2014
                                <1>       ; 04/12/2013
                                <1>       ;
                                <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                                <1>       ;
                                <1> ;-----------------------------------------------
                                <1> ; SCROLL UP
```

```
                        <1> ;     THIS ROUTINE MOVES A BLOCK OF CHARACTERS UP
                        <1> ;     ON THE SCREEN
                        <1> ; INPUT
                        <1> ;     (AH) = CURRENT CRT MODE
                        <1> ;     (AL) = NUMBER OF ROWS TO SCROLL
                        <1> ;     (CX) = ROW/COLUMN OF UPPER LEFT CORNER
                        <1> ;     (DX) = ROW/COLUMN OF LOWER RIGHT CORNER
                        <1> ;     (BH) = ATTRIBUTE TO BE USED ON BLANKED LINE
                        <1> ;     (DS) = DATA SEGMENT
                        <1> ;     (ES) = REGEN BUFFER SEGMENT
                        <1> ; OUTPUT
                        <1> ;     NONE -- THE REGEN BUFFER IS MODIFIED
                        <1> ;--------------------------------------------
                        <1>
                        <1>       ; 07/07/2016
00001F9A 38F5           <1>       cmp    ch, dh
00001F9C 0F8747FBFFFF   <1>       ja     VIDEO_RETURN
00001FA2 38D1           <1>       cmp    cl, dl
00001FA4 0F873FFBFFFF   <1>       ja     VIDEO_RETURN
                        <1>       ;
00001FAA E805000000     <1>       call   _scroll_up
00001FAF E935FBFFFF     <1>         jmp     VIDEO_RETURN
                        <1>
                        <1> _scroll_up:  ; from 'write_tty'
                        <1>       ;
                        <1>       ; cl = left upper column
                        <1>       ; ch = left upper row
                        <1>       ; dl = right lower column
                        <1>       ; dh = right lower row
                        <1>       ;
                        <1>       ; al = line count
                        <1>       ; bl = attribute to be used on blanked line
                        <1>       ; bh = video page number (0 to 7)
                        <1>
00001FB4 E89B000000     <1>       call   test_line_count ; 16/01/2016
                        <1>
00001FB9 8A25[BA6A0000] <1>       mov    ah, [CRT_MODE] ; current video mode
                        <1>       ;;cmp  byte [CRT_MODE], 4
                        <1>       ;cmp   ah, 4 ; 07/07/2016
                        <1>       ;jnb   GRAPHICS_UP ; 26/06/2016
                        <1>       ; 18/11/2020
00001FBF 80FC04         <1>       cmp    ah, 4
00001FC2 720A           <1>       jb     short n0
00001FC4 80FC07         <1>       cmp    ah, 7 ; TEST FOR BW CARD
                        <1>                    ; (80x25 text, mono)
00001FC7 7405           <1>       je     short n0 ; same with mode 3 for TRDOS 386
00001FC9 E92B050000     <1>       jmp    GRAPHICS_UP
                        <1> n0:
                        <1>       ; 07/07/2016
00001FCE 80FF07         <1>       cmp    bh, 7 ; video page number
00001FD1 7606           <1>       jna    short n1
00001FD3 8A3D[D6800100] <1>       mov    bh, [ACTIVE_PAGE]
                        <1> n1:
00001FD9 88DC           <1>       mov    ah, bl ; attribute
                        <1>       ;push  ax ; *
                        <1>       ; 12/04/2021
00001FDB 50             <1>       push   eax
                        <1>       ;mov   esi, [CRT_BASE]
00001FDC BE00800B00     <1>       mov   esi, 0B8000h
00001FE1 3A3D[D6800100] <1>       cmp   bh, [ACTIVE_PAGE]
00001FE7 750B           <1>       jne    short n2
                        <1>       ;
00001FE9 66A1[C4800100] <1>       mov   ax, [CRT_START]
00001FEF 6601C6         <1>       add   si, ax
00001FF2 EB11           <1>       jmp   short n4
                        <1> n2:
00001FF4 20FF           <1>       and   bh, bh
00001FF6 740D           <1>       jz     short n4
00001FF8 88F8           <1>       mov    al, bh
                        <1> n3:
00001FFA 660335[408D0100] <1>     add    si, [CRT_LEN]
00002001 FEC8           <1>       dec    al
00002003 75F5           <1>       jnz    short n3
                        <1> n4:
00002005 E85B000000     <1>       call   scroll_position ; 16/01/2016
0000200A 7420           <1>       jz    short n6
                        <1>
0000200C 01CE           <1>       add   esi, ecx ; from address for scroll
0000200E 88F5           <1>       mov    ch, dh  ; #rows in block
00002010 28C5           <1>       sub    ch, al ; #rows to be moved
                        <1> n5:
00002012 E88B000000     <1>       call   n10 ; 16/01/2016
                        <1>
00002017 51             <1>       push   ecx
00002018 0FB60D[BC6A0000] <1>     movzx  ecx, byte [CRT_COLS]
0000201F 00C9           <1>       add    cl, cl
00002021 01CE           <1>       add    esi, ecx  ; next line
00002023 01CF           <1>       add    edi, ecx
00002025 59             <1>       pop    ecx
                        <1>
00002026 FECD           <1>       dec    ch      ; count of lines to move
00002028 75E8           <1>       jnz    short n5 ; row loop
                        <1>       ; ch = 0
0000202A 88C6           <1>       mov    dh, al  ; #rows
                        <1> n6:
                        <1>       ; attribute in ah
0000202C B020           <1>       mov    al, ' '       ; fill with blanks
                        <1> n7:
0000202E E87C000000     <1>       call   n11 ; 16/01/2016
                        <1>
00002033 8A0D[BC6A0000] <1>       mov    cl, [CRT_COLS]
00002039 00C9           <1>       add    cl, cl
0000203B 01CF           <1>       add    edi, ecx
                        <1>
0000203D FECE           <1>       dec    dh
```

```
5413 0000203F 75ED              <1>       jnz    short n7
5414                            <1> n16:
5415 00002041 3A3D[D6800100]    <1>       cmp    bh, [ACTIVE_PAGE]
5416 00002047 750A              <1>       jne    short n8
5417                            <1>
5418                            <1>       ;cmp   byte [CRT_MODE], 7 ; is this the black and white card
5419                            <1>       ;je    short n8        ; if so, skip the mode reset
5420                            <1>
5421 00002049 A0[BB6A0000]      <1>       mov    al, [CRT_MODE_SET] ; get the value of mode set
5422 0000204E 66BAD803          <1>       mov    dx, 03D8h ; always set color card port
5423 00002052 EE                <1>       out    dx, al
5424                            <1> n8:
5425 00002053 C3                <1>       retn
5426                            <1>
5427                            <1> test_line_count:
5428                            <1>       ; 12/04/2021
5429                            <1>       ; 12/05/2016
5430                            <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
5431                            <1>       ; 07/09/2014 (scroll_up)
5432 00002054 08C0              <1>       or     al, al
5433 00002056 740C              <1>       jz     short al_set2
5434                            <1>       ;push  dx
5435                            <1>       ; 12/04/2021
5436 00002058 52                <1>       push   edx
5437 00002059 28EE              <1>       sub    dh, ch  ; subtract upper row from lower row number
5438 0000205B FEC6              <1>       inc    dh     ; adjust difference by 1
5439 0000205D 38C6              <1>       cmp    dh, al       ; line count = amount of rows in window?
5440 0000205F 7502              <1>       jne    short al_set1 ; if not the we're all set
5441 00002061 30C0              <1>       xor    al, al ; otherwise set al to zero
5442                            <1> al_set1:
5443                            <1>       ;pop   dx
5444                            <1>       ; 12/04/2021
5445 00002063 5A                <1>       pop    edx
5446                            <1> al_set2:
5447 00002064 C3                <1>       retn
5448                            <1>
5449                            <1> scroll_position:
5450                            <1>       ; 12/04/2021
5451                            <1>       ; 26/06/2016
5452                            <1>       ; 30/01/2016
5453                            <1>         ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
5454                            <1>       ; 07/09/2014 (scroll_up)
5455                            <1>
5456                            <1>       ; (*) [esp+4] = ax (al = line count, ah = attribute)
5457                            <1>
5458                            <1>       ;push  dx
5459                            <1>       ; 12/04/2021
5460 00002065 52                <1>       push   edx
5461 00002066 6689CA            <1>       mov    dx, cx ; now, upper left position in DX
5462 00002069 E8F1FEFFFF        <1>       call   position
5463 0000206E 01C6              <1>       add    esi, eax
5464 00002070 89F7              <1>       mov    edi, esi
5465                            <1>       ;pop   dx     ; lower right position in DX
5466                            <1>       ; 12/04/2021
5467 00002072 5A                <1>       pop    edx
5468 00002073 6629CA            <1>       sub    dx, cx
5469 00002076 FEC6              <1>       inc    dh     ; dh = #rows
5470 00002078 FEC2              <1>       inc    dl     ; dl = #cols in block
5471 0000207A 59                <1>       pop    ecx    ; return address
5472                            <1>       ;pop   ax     ; * ; al = line count, ah = attribute
5473                            <1>       ; 12/04/2021
5474 0000207B 58                <1>       pop    eax ; (*)
5475 0000207C 51                <1>       push   ecx    ; return address
5476 0000207D 0FB7C8            <1>       movzx  ecx, ax
5477 00002080 8A25[BC6A0000]    <1>       mov    ah, [CRT_COLS]
5478 00002086 F6E4              <1>       mul    ah     ; determine offset to from address
5479 00002088 6601C0            <1>       add    ax, ax  ; *2 for attribute byte
5480                            <1>       ;
5481                            <1>       ;push  ax     ; offset
5482                            <1>       ;push  dx
5483                            <1>       ; 12/04/2021
5484 0000208B 50                <1>       push   eax    ; offset
5485 0000208C 52                <1>       push   edx
5486                            <1>       ;
5487                            <1>       ; 04/04/2014
5488 0000208D 66BADA03          <1>       mov    dx, 3DAh ; guaranteed to be color card here
5489                            <1> n9:                 ; wait_display_enable
5490 00002091 EC                <1>       in    al, dx   ; get port
5491 00002092 A808              <1>       test   al, RVRT ; wait for vertical retrace
5492 00002094 74FB              <1>       jz     short n9 ; wait_display_enable
5493 00002096 B025              <1>       mov    al, 25h
5494 00002098 B2D8              <1>       mov    dl, 0D8h ; address control port
5495 0000209A EE                <1>       out    dx, al ; turn off video during vertical retrace
5496                            <1>       ;pop   dx     ; #rows, #cols
5497                            <1>       ;pop   ax     ; offset
5498                            <1>       ; 12/04/2021
5499 0000209B 5A                <1>       pop    edx    ; #rows, #cols
5500 0000209C 58                <1>       pop    eax    ; offset
5501 0000209D 6691              <1>       xchg   ax, cx ;
5502                            <1>       ; ecx = offset, al = line count, ah = attribute
5503                            <1>       ;
5504 0000209F 08C0              <1>       or     al, al
5505 000020A1 C3                <1>       retn
5506                            <1> n10:
5507                            <1>       ; Move rows
5508 000020A2 88D1              <1>       mov    cl, dl ; get # of cols to move
5509 000020A4 56                <1>       push   esi
5510 000020A5 57                <1>       push   edi    ; save start address
5511                            <1> n10r:
5512 000020A6 66A5              <1>       movsw         ; move that line on screen
5513 000020A8 FEC9              <1>       dec    cl
5514 000020AA 75FA              <1>       jnz   short n10r
5515 000020AC 5F                <1>       pop    edi
5516 000020AD 5E                <1>       pop    esi    ; recover addresses
5517 000020AE C3                <1>       retn
```

```
5518                                    <1> n11:
5519                                    <1>       ; Clear rows
5520                                    <1>                       ; dh =  #rows
5521 000020AF 88D1                     <1>       mov    cl, dl ; get # of cols to clear
5522 000020B1 57                       <1>       push   edi     ; save address
5523                                    <1> n11r:
5524 000020B2 66AB                     <1>       stosw          ; store fill character
5525 000020B4 FEC9                     <1>       dec    cl
5526 000020B6 75FA                     <1>       jnz    short n11r
5527 000020B8 5F                       <1>       pop    edi     ; recover address
5528 000020B9 C3                       <1>       retn
5529                                    <1>
5530                                    <1> SCROLL_DOWN:
5531                                    <1>       ; 12/04/2021
5532                                    <1>       ; 07/07/2016
5533                                    <1>       ; 27/06/2016
5534                                    <1>       ; 26/06/2016
5535                                    <1>       ; 12/05/2016
5536                                    <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
5537                                    <1>       ;
5538                                    <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
5539                                    <1>
5540                                    <1> ;--------------------------------------------
5541                                    <1> ; SCROLL DOWN
5542                                    <1> ;     THIS ROUTINE MOVES THE CHARACTERS WITHIN A DEFINED
5543                                    <1> ;     BLOCK DOWN ON THE SCREEN, FILLING THE TOP LINES
5544                                    <1> ;     WITH A DEFINED CHARACTER
5545                                    <1> ; INPUT
5546                                    <1> ;     (AH) = CURRENT CRT MODE
5547                                    <1> ;     (AL) = NUMBER OF LINES TO SCROLL
5548                                    <1> ;     (CX) = UPPER LEFT CORNER OF RECION
5549                                    <1> ;     (DX) = LOWER RIGHT CORNER OF REGION
5550                                    <1> ;     (BH) = FILL CHARACTER
5551                                    <1> ;     (DS) = DATA SEGMENT
5552                                    <1> ;     (ES) = REGEN SEGMENT
5553                                    <1> ; OUTPUT
5554                                    <1> ;     NONE -- SCREEN IS SCROLLED
5555                                    <1> ;--------------------------------------------
5556                                    <1>
5557                                    <1>       ; 07/07/2016
5558 000020BA 38F5                     <1>       cmp    ch, dh
5559                                    <1>       ;ja    VIDEO_RETURN
5560 000020BC 7709                     <1>       ja     short _s_d_retn ; 18/11/2020
5561 000020BE 38D1                     <1>       cmp    cl, dl
5562                                    <1>       ;ja    VIDEO_RETURN
5563 000020C0 7705                     <1>       ja     short _s_d_retn ; 18/11/2020
5564                                    <1>       ;
5565 000020C2 E805000000               <1>       call   _scroll_down
5566                                    <1> _s_d_retn:
5567 000020C7 E91DFAFFFF               <1>       jmp     VIDEO_RETURN
5568                                    <1>
5569                                    <1> _scroll_down: ; 27/06/2016
5570                                    <1>
5571                                    <1>       ; cl = left upper column
5572                                    <1>       ; ch = left upper row
5573                                    <1>       ; dl = right lower column
5574                                    <1>       ; dh = right lower row
5575                                    <1>       ;
5576                                    <1>       ; al = line count
5577                                    <1>       ; bl = attribute to be used on blanked line
5578                                    <1>       ; bh = video page number (0 to 7)
5579                                    <1>
5580                                    <1>       ; !!!!
5581 000020CC FD                       <1>       std    ; DIRECTION FOR SCROLL DOWN
5582                                    <1>       ; !!!!
5583 000020CD E882FFFFFF               <1>       call   test_line_count ; 16/01/2016
5584                                    <1>
5585 000020D2 8A25[BA6A0000]           <1>       mov    ah, [CRT_MODE] ; current video mode
5586                                    <1>       ;;cmp  byte [CRT_MODE], 4
5587                                    <1>       ;cmp   ah, 4 ; 07/07/2016
5588                                    <1>       ;jnb   GRAPHICS_DOWN ; 26/06/2016
5589                                    <1>       ; 18/11/2020
5590 000020D8 80FC04                   <1>       cmp    ah, 4
5591 000020DB 720A                     <1>       jb     short _n0
5592 000020DD 80FC07                   <1>       cmp    ah, 7 ; TEST FOR BW CARD
5593                                    <1>                    ;      (80x25 text, mono)
5594 000020E0 7405                     <1>       je     short _n0 ; same with mode 3 for TRDOS 386
5595 000020E2 E9EE070000               <1>       jmp    GRAPHICS_DOWN
5596                                    <1> _n0:
5597                                    <1>       ; 07/07/2016
5598 000020E7 80FF07                   <1>       cmp    bh, 7 ; video page number
5599 000020EA 7606                     <1>       jna    short n12
5600 000020EC 8A3D[D6800100]           <1>       mov    bh, [ACTIVE_PAGE]
5601                                    <1>       ;
5602                                    <1> n12:          ; CONTINUE_DOWN
5603 000020F2 88DC                     <1>       mov    ah, bl
5604                                    <1>       ;push  ax     ; * ; save attribute in ah
5605                                    <1>       ; 12/04/2021
5606 000020F4 50                       <1>       push   eax
5607 000020F5 6689D0                   <1>       mov    ax, dx ; LOWER RIGHT CORNER
5608 000020F8 E868FFFFFF               <1>       call   scroll_position     ; GET REGEN LOCATION
5609 000020FD 741F                     <1>       jz     short n14
5610 000020FF 29CE                     <1>       sub    esi, ecx  ; SI IS FROM ADDRESS
5611 00002101 88F5                     <1>       mov    ch, dh  ; #rows in block
5612 00002103 28C5                     <1>       sub    ch, al ; #rows to be moved
5613                                    <1> n13:
5614 00002105 E898FFFFFF               <1>       call   n10    ; MOVE ONE ROW
5615                                    <1>
5616 0000210A 51                       <1>       push   ecx
5617 0000210B 8A0D[BC6A0000]           <1>       mov    cl, [CRT_COLS]
5618 00002111 00C9                     <1>       add    cl, cl
5619 00002113 29CE                     <1>       sub    esi, ecx  ; next line
5620 00002115 29CF                     <1>       sub    edi, ecx
5621 00002117 59                       <1>       pop    ecx
5622                                    <1>
```

```
                                <1>       dec    ch      ; count of lines to move
                                <1>       jnz    short n13 ; row loop
                                <1>       ; ch = 0
                                <1>       mov    dh, al  ; #rows
                                <1> n14:
                                <1>       ; attribute in ah
                                <1>       mov    al, ' '       ; fill with blanks
                                <1> n15:
                                <1>       call   n11 ; 16/01/2016
                                <1>
                                <1>       mov    cl, [CRT_COLS]
                                <1>       add    cl, cl
                                <1>       sub    edi, ecx
                                <1>
                                <1>       dec    dh
                                <1>       jnz    short n15
                                <1>       ;
                                <1>       ; 18/11/2020
                                <1>       cld    ; clear direction flag
                                <1>       ;
                                <1>       jmp    n16 ; 27/06/2016
                                <1>
                                <1> ;     cmp    bh, [ACTIVE_PAGE]
                                <1> ;     jne    short n16
                                <1> ;
                                <1> ;     ;cmp   byte [CRT_MODE], 7  ; is this the black and white card
                                <1> ;     ;je    short n16           ; if so, skip the mode reset
                                <1> ;
                                <1> ;     mov    al, [CRT_MODE_SET] ; get the value of mode set
                                <1> ;     mov    dx, 03D8h ; always set color card port
                                <1> ;     out    dx, al
                                <1> ;n16:
                                <1> ;     ; !!!!
                                <1> ;     cld          ; Clear direction flag !
                                <1> ;     ; !!!!
                                <1> ;     retn
                                <1>
                                <1> READ_AC_CURRENT:
                                <1>       ; 08/07/2016
                                <1>       ; 26/06/2016
                                <1>       ; 12/05/2016
                                <1>       ; 18/01/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ;
                                <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                                <1>       ;
                                <1>       ; 08/07/2016
00002139 803D[BA6A0000]07       <1>       cmp    byte [CRT_MODE], 7 ; 6!?
00002140 7607                   <1>       jna    short read_ac_c
00002142 31C0                   <1>       xor    eax, eax
00002144 E9A5F9FFFF             <1>       jmp   _video_return
                                <1> read_ac_c:
00002149 E805000000             <1>       call   _read_ac_current
                                <1>       ; 12/05/2016
                                <1>       ;jmp   VIDEO_RETURN
0000214E E99BF9FFFF             <1>       jmp    _video_return
                                <1>
                                <1> ;--------------------------------------------------------------------------
                                <1> ; READ_AC_CURRENT                                                       :
                                <1> ;     THIS ROUTINE READS THE ATTRIBUTE AND CHARACTER AT THE CURRENT      :
                                <1> ;     CURSOR POSITION AND RETURNS THEM TO THE CALLER                    :
                                <1> ; INPUT                                                                 :
                                <1> ;     (AH) = CURRENT CRT MODE                                           :
                                <1> ;     (BH) = DISPLAY PAGE ( ALPHA MODES ONLY )                          :
                                <1> ;     (DS) = DATA SEGMENT                                               :
                                <1> ;     (ES) = REGEN SEGMENT                                              :
                                <1> ; OUTPUT                                                                :
                                <1> ;     (AL) = CHARACTER READ                                             :
                                <1> ;     (AH) = ATTRIBUTE READ                                             :
                                <1> ;--------------------------------------------------------------------------
                                <1>
                                <1>  read_ac_current:
                                <1>       ; 26/06/2016
                                <1>       ; 12/05/2016
                                <1>       ; 18/01/2016
                                <1>
00002153 8A25[BA6A0000]         <1>       mov    ah, [CRT_MODE] ; current video mode
00002159 80FC04                 <1>       cmp    ah, 4
0000215C 720A                   <1>       jb     short p10
                                <1>       ; 18/11/2020
                                <1>       ;cmp   byte [CRT_MODE], 4
                                <1>       ;jnb   GRAPHICS_READ ; 26/06/2016
                                <1>
0000215E 80FC07                 <1>       cmp    ah, 7 ; TEST FOR BW CARD (80x25 monochrome text)
00002161 7405                   <1>       je     short p10     ; same with mode 3 in TRDOS 386
00002163 E9C3080000             <1>       jmp    GRAPHICS_READ
                                <1> p10:
00002168 E801FEFFFF             <1>       call   find_position; GET REGEN LOCATION AND PORT ADDRESS
                                <1>       ;
                                <1>       ; esi = regen location
                                <1>       ; dx = status port
                                <1>       ;
                                <1>
                                <1>       ; 18/11/2020
                                <1>       ; convert display mode to a zero value
                                <1>       ; for 80 column color mode
                                <1>       ;mov   ah, [CRT_MODE]
                                <1>       ;sub   ah, 2
                                <1>       ;shr   ah, 1
                                <1>       ;jnz   short p13
                                <1>
                                <1>       ; 05/12/2020
                                <1>       ; 18/11/2020 (TRDOS 386 v2.0.3)
                                <1>       ;xor   bl, bl ; 0
0000216D 803D[BA6A0000]03       <1>       cmp    byte [CRT_MODE], 03h ; 80x25 color text
```

```
5728                                    <1>       ;je    short p11    ; Note: Only mode 03h and mode 01h are
5729                                    <1>       ;inc   bl     ; 1    ;      in use by TRDOS 386 as text modes
5730                                    <1>       ;jmp   short p14    ;      (07h, 00h and 02h are redirected)
5731 00002174 7516                     <1>       jne    short p13
5732                                    <1>
5733                                    <1>       ; 05/12/2020
5734 00002176 3A3D[D6800100]           <1>       cmp    bh, [ACTIVE_PAGE]
5735 0000217C 750E                     <1>       jne    short p13
5736                                    <1>
5737                                    <1>       ; WAIT FOR HORIZONTAL RETRACE OR VERTICAL RETRACE IF COLOR 80
5738                                    <1> p11:
5739 0000217E FB                       <1>       sti           ; enable interrupts first
5740                                    <1>       ; 05/12/2020
5741 0000217F 90                       <1>       nop
5742                                    <1>       ; 18/11/2020
5743                                    <1>       ;or    bl, bl
5744                                    <1>       ;jnz   short p13 ; mode 01h (and mode 00h) - 40x25 color text
5745 00002180 FA                       <1>       cli           ; block interrupts for single loop
5746 00002181 EC                       <1>       in     al, dx ; get status from the adapter
5747 00002182 A801                     <1>       test   al, RHRZ ; is horizontal retrace low
5748 00002184 75F8                     <1>       jnz    short p11 ; wait until it is
5749                                    <1> p12:                ;  wait for either retrace high
5750 00002186 EC                       <1>       in     al, dx ; get status again
5751 00002187 A809                     <1>       test   al, RVRT+RHRZ ; is horizontal or vertical retrace high
5752 00002189 74FB                     <1>       jz     short p12 ; wait until either retrace active
5753                                    <1> ;p14:
5754 0000218B FB                       <1>       sti
5755                                    <1> p13:
5756 0000218C 81C600800B00             <1>       add    esi, 0B8000h
5757 00002192 668B06                   <1>       mov    ax, [esi]
5758                                    <1>
5759 00002195 C3                       <1>       retn   ; 18/01/2016
5760                                    <1>
5761                                    <1> WRITE_AC_CURRENT:
5762                                    <1>       ; 08/07/2016
5763                                    <1>       ; 26/06/2016
5764                                    <1>       ; 24/06/2016
5765                                    <1>       ; 12/05/2016
5766                                    <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
5767                                    <1>       ;
5768                                    <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
5769                                    <1>       ;
5770                                    <1> ;--------------------------------------------------------------------
5771                                    <1> ; WRITE_AC_CURRENT                                                 :
5772                                    <1> ;     THTS ROUTINE WRITES THE ATTRIBUTE AND CHARACTER               :
5773                                    <1> ;     AT THE CURRENT CURSOR POSITION                               :
5774                                    <1> ; INPUT                                                            :
5775                                    <1> ;     (AH) = CURRENT CRT MODE                                      :
5776                                    <1> ;     (BH) = DISPLAY PAGE                                          :
5777                                    <1> ;     (CX) = COUNT OF CHARACTERS TO WRITE                          :
5778                                    <1> ;     (AL) = CHAR TO WRITE                                         :
5779                                    <1> ;     (BL) = ATTRIBUTE OF CHAR TO WRITE                            :
5780                                    <1> ;     (DS) = DATA SEGMENT                                          :
5781                                    <1> ;     (ES) = REGEN SEGMENT                                         :
5782                                    <1> ; OUTPUT                                                           :
5783                                    <1> ;     DISPLAY REGEN BUFFER UPDATED                                 :
5784                                    <1> ;--------------------------------------------------------------------
5785                                    <1>
5786                                    <1>       ; 08/07/2016
5787 00002196 803D[BA6A0000]07         <1>       cmp    byte [CRT_MODE], 7 ; 6!?
5788 0000219D 760A                     <1>       jna    short write_ac_c
5789                                    <1>
5790 0000219F E8F50A0000               <1>       call   vga_write_char_attr
5791 000021A4 E940F9FFFF               <1>       jmp    VIDEO_RETURN
5792                                    <1>
5793                                    <1> write_ac_c:
5794 000021A9 E834000000               <1>       call   _write_c_current
5795                                    <1>
5796 000021AE 0FB6F7                   <1>       movzx  esi, bh ; video page number (0 to 7)
5797 000021B1 889E[C36A0000]           <1>       mov    [esi+chr_attrib], bl ; color/attribute
5798                                    <1>
5799 000021B7 E92DF9FFFF               <1>       jmp    VIDEO_RETURN
5800                                    <1>
5801                                    <1> WRITE_C_CURRENT:
5802                                    <1>       ; 08/07/2016
5803                                    <1>       ; 26/06/2016
5804                                    <1>       ; 12/05/2016
5805                                    <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
5806                                    <1>       ;
5807                                    <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
5808                                    <1>       ;
5809                                    <1> ;--------------------------------------------------------------------
5810                                    <1> ; WRITE_C_CURRENT                                                  :
5811                                    <1> ;     THIS ROUTINE WRITES THE CHARACTER AT                         :
5812                                    <1> ;     THE CURRENT CURSOR POSITION, ATTRIBUTE UNCHANGED              :
5813                                    <1> ; INPUT                                                            :
5814                                    <1> ;     (AH) = CURRENT CRT MODE                                      :
5815                                    <1> ;     (BH) = DISPLAY PAGE                                          :
5816                                    <1> ;     (CX) = COUNT OF CHARACTERS TO WRITE                          :
5817                                    <1> ;     (AL) = CHAR TO WRITE                                         :
5818                                    <1> ;     (DS) = DATA SEGMENT                                          :
5819                                    <1> ;     (ES) = REGEN SEGMENT                                         :
5820                                    <1> ; OUTPUT                                                           :
5821                                    <1> ;     DISPLAY REGEN BUFFER UPDATED                                 :
5822                                    <1> ;--------------------------------------------------------------------
5823                                    <1>
5824                                    <1>       ; 08/07/2016
5825 000021BC 803D[BA6A0000]07         <1>       cmp    byte [CRT_MODE], 7 ; 6!?
5826 000021C3 760A                     <1>       jna    short write_c_c
5827                                    <1>
5828 000021C5 E8CF0A0000               <1>       call   vga_write_char_only
5829 000021CA E91AF9FFFF               <1>       jmp    VIDEO_RETURN
5830                                    <1>
5831                                    <1> write_c_c:
5832                                    <1>       ;and   bh, 7 ; video page number (<= 7)
```

```
5833 000021CF 0FB6F7              <1>       movzx  esi, bh
5834 000021D2 8A9E[C36A0000]      <1>       mov    bl, [esi+chr_attrib]
5835                              <1>
5836 000021D8 E805000000          <1>       call   _write_c_current
5837 000021DD E907F9FFFF          <1>       jmp    VIDEO_RETURN
5838                              <1>
5839                              <1> _write_c_current:  ; from 'write_tty'
5840                              <1>       ; 12/04/2021
5841                              <1>       ; 18/11/2020
5842                              <1>       ; 26/06/2016
5843                              <1>       ; 24/06/2016
5844                              <1>       ; 12/05/2016
5845                              <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
5846                              <1>       ; 30/08/2014 (Retro UNIX 386 v1)
5847                              <1>       ; 18/01/2014
5848                              <1>       ; 04/12/2013
5849                              <1>       ;
5850                              <1>       ; VIDEO.ASM - 06/10/85 VIDEO DISPLAY BIOS
5851                              <1>
5852                              <1>       ; 18/11/2020
5853 000021E2 8A25[BA6A0000]      <1>       mov    ah, [CRT_MODE] ; current video mode
5854 000021E8 80FC04              <1>       cmp    ah, 4
5855 000021EB 720A                <1>       jb     short p40
5856                              <1>
5857                              <1>       ;cmp   byte [CRT_MODE], 4
5858                              <1>       ;jnb   GRAPHICS_WRITE ; 26/06/2016
5859                              <1>
5860 000021ED 80FC07              <1>       cmp    ah, 7 ; TEST FOR BW CARD
5861 000021F0 7405                <1>       je     short p40
5862 000021F2 E984070000          <1>       jmp    GRAPHICS_WRITE
5863                              <1> p40:
5864                              <1>       ; al = character
5865                              <1>       ; bl = color/attribute
5866                              <1>       ; bh = video page
5867                              <1>       ; cx = count of characters to write
5868                              <1>       ;push  dx
5869                              <1>       ; 12/04/2021
5870 000021F7 52                  <1>       push   edx ; *
5871 000021F8 88DC                <1>       mov    ah, bl  ; color/attribute (12/05/2016)
5872                              <1>       ;push  ax     ; save character & attribute/color
5873                              <1>       ;push  cx
5874                              <1>       ; 12/04/2021
5875 000021FA 50                  <1>       push   eax ; ** ; save character & attribute/color
5876 000021FB 51                  <1>       push   ecx ; ***
5877 000021FC E86DFDFFFF          <1>       call   find_position  ; get regen location and port address
5878                              <1>       ;pop   cx
5879                              <1>       ; 12/04/2021
5880 00002201 59                  <1>       pop    ecx ; ***
5881                              <1>       ; esi = regen location
5882                              <1>       ; dx = status port
5883                              <1>       ;
5884 00002202 81C600800B00        <1>       add    esi, 0B8000h ; 30/08/2014 (crt_base)
5885                              <1>       ;
5886                              <1>       ; 18/11/2020
5887                              <1>       ; convert display mode to a zero value
5888                              <1>       ; for 80 column color mode
5889                              <1>       ;mov   ah, [CRT_MODE]
5890                              <1>       ;sub   ah, 2
5891                              <1>       ;shr   ah, 1
5892                              <1>       ;jnz   short p44    ; 26/06/2016
5893                              <1>
5894                              <1>       ; 05/12/2020
5895                              <1>       ; 18/11/2020 (TRDOS 386 v2.0.3)
5896                              <1>       ;xor   bl, bl ; 0
5897 00002208 803D[BA6A0000]03    <1>       cmp    byte [CRT_MODE], 03h ; 80x25 color text
5898                              <1>       ;je    short p41    ; Note: Only mode 03h and mode 01h are
5899                              <1>       ;inc   bl     ; 1    ;      in use by TRDOS 386 as text modes
5900                              <1>       ;jmp   short p43    ;      (07h, 00h and 02h are redirected)
5901                              <1>       ; 05/12/2020
5902 0000220F 751A                <1>       jne    short p44
5903                              <1> p46:
5904                              <1>       ; 05/12/2020
5905 00002211 3A3D[D6800100]      <1>       cmp     bh, [ACTIVE_PAGE]
5906 00002217 7512                <1>       jne    short p44
5907                              <1>
5908                              <1>       ; WAIT FOR HORIZONTAL RETRACE OR VERTICAL RETRACE IF COLOR 80
5909                              <1> p41:
5910                              <1>       ; 05/12/2020
5911 00002219 FB                  <1>       sti           ; enable interrupts first
5912 0000221A 90                  <1>       nop
5913                              <1>       ; 18/11/2020
5914                              <1>       ;or    bl, bl
5915                              <1>       ;jnz   short p44 ; mode 01h (and mode 00h) - 40x25 color text
5916 0000221B FA                  <1>       cli           ; block interrupts for single loop
5917 0000221C EC                  <1>       in     al, dx ; get status from the adapter
5918 0000221D A808                <1>       test   al, RVRT ; check for vertical retrace first
5919 0000221F 7509                <1>       jnz    short p43 ; Do fast write now if vertical retrace
5920 00002221 A801                <1>       test   al, RHRZ ; is horizontal retrace low
5921 00002223 75F4                <1>       jnz    short p41 ; wait until it is
5922                              <1> p42:                ;  wait for either retrace high
5923 00002225 EC                  <1>       in     al, dx ; get status again
5924 00002226 A809                <1>       test   al, RVRT+RHRZ ; is horizontal or vertical retrace high
5925 00002228 74FB                <1>       jz     short p42 ; wait until either retrace active
5926                              <1> p43:
5927 0000222A FB                  <1>       sti
5928                              <1> p44:
5929 0000222B 668B0424            <1>       mov    ax, [esp] ; restore the character (al) & attribute (ah)
5930 0000222F 668906              <1>       mov    [esi], ax
5931                              <1>
5932 00002232 6649                <1>       dec    cx
5933 00002234 740D                <1>       jz     short p45
5934                              <1>
5935 00002236 46                  <1>       inc    esi
5936 00002237 46                  <1>       inc    esi
5937                              <1>
```

```
                                <1>       ; 05/12/2020
00002238 803D[BA6A0000]03       <1>       cmp    byte [CRT_MODE], 03h
0000223F 75EA                   <1>       jne    short p44
                                <1>       ;jmp   short p41
00002241 EBCE                   <1>       jmp    short p46
                                <1> p45:
                                <1>       ;pop   ax
                                <1>       ;pop   dx
                                <1>       ; 12/04/2021
00002243 58                     <1>       pop    eax ; **
00002244 5A                     <1>       pop    edx ; *
00002245 C3                     <1>       retn
                                <1>
                                <1> ; 09/07/2016
                                <1> ; 26/06/2016
                                <1> ; 24/06/2016
                                <1> ; 12/05/2016
                                <1> ; 18/01/2016
                                <1> ; 16/01/2016 - TRDOS 386 (TRDOS v2.0)
                                <1> ; 30/06/2015
                                <1> ; 27/06/2015
                                <1> ; 11/03/2015
                                <1> ; 02/09/2014
                                <1> ; 30/08/2014
                                <1> ; VIDEO FUNCTIONS
                                <1> ; (write_tty - Retro UNIX 8086 v1 - U9.ASM, 01/02/2014)
                                <1>
                                <1> WRITE_TTY:
                                <1>       ; 09/12/2017
                                <1>       ; 09/07/2016
                                <1>       ; 01/07/2016
                                <1>       ; 26/06/2016
                                <1>       ; 24/06/2016
                                <1>       ; 13/05/2016
                                <1>       ; 12/05/2016
                                <1>       ; 30/01/2016
                                <1>       ; 18/01/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 13/08/2015
                                <1>       ; 02/09/2014
                                <1>       ; 30/08/2014 (Retro UNIX 386 v1 - beginning)
                                <1>       ; 01/02/2014 (Retro UNIX 8086 v1 - last update)
                                <1>       ; 03/12/2013 (Retro UNIX 8086 v1 - beginning)
                                <1>       ; (Modified registers: EAX, EBX, ECX, EDX, ESI, EDI)
                                <1>       ;
                                <1>       ; INPUT -> AL = Character to be written
                                <1>       ;         BL = Color (Forecolor, Backcolor)
                                <1>       ;         BH = Video Page (0 to 7)
                                <1>       ;
                                <1>       ; 09/07/2016
00002246 803D[BA6A0000]07       <1>       cmp    byte [CRT_MODE], 7
0000224D 760A                   <1>       jna    short write_tty_cga
                                <1>
0000224F E8300D0000             <1>       call   vga_write_teletype
00002254 E990F8FFFF             <1>       jmp    VIDEO_RETURN
                                <1>
                                <1> write_tty_cga:
                                <1>       ; 13/05/2016
                                <1>       ;call  _write_tty
                                <1>       ; 01/07/2016
00002259 E818000000             <1>       call   _write_tty_m3
0000225E E986F8FFFF             <1>       jmp    VIDEO_RETURN
                                <1>
                                <1> RVRT  equ    00001000b    ; VIDEO VERTICAL RETRACE BIT
                                <1> RHRZ  equ    00000001b    ; VIDEO HORIZONTAL RETRACE BIT
                                <1>
                                <1> ; Derived from "WRITE_TTY" procedure of IBM "pc-at" rombios source code
                                <1> ; (06/10/1985), 'video.asm', INT 10H, VIDEO_IO
                                <1> ;
                                <1> ; 06/10/85  VIDEO DISPLAY BIOS
                                <1> ;
                                <1> ;--- WRITE_TTY --------------------------------------------------------------------
                                <1> ;                                                                               :
                                <1> ;   THIS INTERFACE PROVIDES A TELETYPE LIKE INTERFACE TO THE                    :
                                <1> ;   VIDEO CARDS. THE INPUT CHARACTER IS WRITTEN TO THE CURRENT                  :
                                <1> ;   CURSOR POSITION, AND THE CURSOR IS MOVED TO THE NEXT POSITION.             :
                                <1> ;   IF THE CURSOR LEAVES THE LAST COLUMN OF THE FIELD, THE COLUMN              :
                                <1> ;   IS SET TO ZERO, AND THE ROW VALUE IS INCREMENTED. IF THE ROW                :
                                <1> ;   ROW VALUE LEAVES THE FIELD, THE CURSOR IS PLACED ON THE LAST ROW,          :
                                <1> ;   FIRST COLUMN, AND THE ENTIRE SCREEN IS SCROLLED UP ONE LINE.               :
                                <1> ;   WHEN THE SCREEN IS SCROLLED UP, THE ATTRIBUTE FOR FILLING THE              :
                                <1> ;   NEWLY BLANKED LINE IS READ FROM THE CURSOR POSITION ON THE PREVIOUS        :
                                <1> ;   LINE BEFORE THE SCROLL, IN CHARACTER MODE. IN GRAPHICS MODE,               :
                                <1> ;   THE 0 COLOR IS USED.                                                      :
                                <1> ;   ENTRY --                                                                  :
                                <1> ;     (AH) = CURRENT CRT MODE                                                 :
                                <1> ;     (AL) = CHARACTER TO BE WRITTEN                                          :
                                <1> ;         NOTE THAT BACK SPACE, CARRIAGE RETURN, BELL AND LINE FEED ARE       :
                                <1> ;         HANDLED AS COMMANDS RATHER THAN AS DISPLAY GRAPHICS CHARACTERS      :
                                <1> ;     (BL) = FOREGROUND COLOR FOR CHAR WRITE IF CURRENTLY IN A GRAPHICS MODE  :
                                <1> ;   EXIT --                                                                   :
                                <1> ;     ALL REGISTERS SAVED                                                     :
                                <1> ;-------------------------------------------------------------------------------
                                <1>
                                <1> ; 18/11/2020 (TRDOS 386 v2.0.3)
                                <1> ; 09/12/2017
                                <1> ; 08/07/2016
                                <1> ; 26/06/2016
                                <1> ; 24/06/2016
                                <1> _write_tty:
                                <1>       ; 13/05/2016
                                <1>       ; --- 18/11/2020 ---
                                <1>       ; NOTE:
                                <1>       ; Only kernel calls "_write_tty" procedure...
                                <1>       ; TRDOS 386 v2 kernel uses video mode 3 for displaying
```

```
                                <1>       ; (some error) messages and also mainprog command interpreter
                                <1>       ; (in kernel) uses "_write_tty".
                                <1>       ; So, here video mode must be set to 3 if it is not 3.
                                <1>
00002263 FA                     <1>       cli    ; disable interrupts
                                <1>       ;
                                <1>       ; 01/09/2014
00002264 803D[BA6A0000]03       <1>       cmp    byte [CRT_MODE], 3
0000226B 7409                   <1>       je     short _write_tty_m3
                                <1>       ;
                                <1> set_mode_3:
0000226D 53                     <1>       push   ebx
0000226E 50                     <1>       push   eax
                                <1>       ;call  _set_mode
                                <1>       ; 18/11/2020
0000226F E884F8FFFF             <1>       call   set_txt_mode  ; set video mode to 03h
00002274 58                     <1>       pop    eax
00002275 5B                     <1>       pop    ebx
                                <1>       ;
                                <1> _write_tty_m3: ; 24/06/2016 (m3 -> _write_tty_m3)
00002276 0FB6F7                 <1>       movzx  esi, bh ; 12/05/2016
00002279 66D1E6                 <1>       shl    si, 1
0000227C 81C6[C6800100]         <1>       add    esi, CURSOR_POSN
00002282 668B16                 <1>       mov    dx, [esi]
                                <1>       ;
                                <1>       ; dx now has the current cursor position
                                <1>       ;
00002285 3C0D                   <1>       cmp    al, 0Dh      ; CR   ; is it carriage return or control character
                                <1>       ;jbe   short u8
                                <1>       ; 17/04/2021
00002287 723F                   <1>       jb     short u8
00002289 746D                   <1>       je     short u9
                                <1>       ;
                                <1>       ; write the char to the screen
                                <1> u0:
                                <1>       ; al = character
                                <1>       ; bl = attribute/color
                                <1>       ; bh = video page number (0 to 7)
                                <1>       ;
0000228B 66B90100               <1>       mov    cx, 1  ; 24/06/2016
                                <1>       ; cx = count of characters to write
                                <1>       ;
0000228F E84EFFFFFF             <1>       call   _write_c_current ; 16/01/2015
                                <1>       ;
                                <1>       ; position the cursor for next char
00002294 FEC2                   <1>       inc    dl           ; next column
00002296 3A15[BC6A0000]         <1>       cmp    dl, [CRT_COLS]  ; test for column overflow
0000229C 755C                   <1>       jne    _set_cpos
0000229E B200                   <1>       mov    dl, 0        ; column = 0
                                <1> u10:                      ; (line feed found)
000022A0 80FE18                 <1>       cmp    dh, 25-1     ; check for last row
000022A3 721F                   <1>       jb     short u6
                                <1>       ;
                                <1>       ; scroll required
                                <1> u1:
                                <1>       ; SET CURSOR POSITION (04/12/2013)
000022A5 E850000000             <1>       call   _set_cpos
                                <1>       ;
                                <1>       ; determine value to fill with during scroll
                                <1> u2:
                                <1>       ; bh = video page number
                                <1>       ;
000022AA E8A4FEFFFF             <1>       call   _read_ac_current ; 18/01/2016
                                <1>       ;
                                <1>       ; al = character, ah = attribute
                                <1>       ; bh = video page number
                                <1>       ; 18/11/2020
000022AF 88E3                   <1>       mov    bl, ah ; color/attribute
                                <1> u3:
                                <1>       ;;mov  ax, 0601h    ; scroll one line
                                <1>       ;;sub  cx, cx       ; upper left corner
                                <1>       ;;mov  dh, 25-1     ; lower right row
                                <1>       ;;;mov dl, [CRT_COLS]
                                <1>       ;mov   dl, 80       ; lower right column
                                <1>       ;;dec  dl
                                <1>       ;;mov  dl, 79
                                <1>
                                <1>       ;;call scroll_up    ; 04/12/2013
                                <1>       ;;; 11/03/2015
                                <1>       ; 02/09/2014
                                <1>       ;;;mov cx, [crt_ulc] ; Upper left corner  (0000h)
                                <1>       ;;;mov dx, [crt_lrc] ; Lower right corner (184Fh)
                                <1>       ; 11/03/2015
                                <1>       ;sub   cx, cx
                                <1>       ; 17/04/2021
000022B1 29C9                   <1>       sub    ecx, ecx
                                <1>       ;mov   dx, 184Fh ; dl = 79 (column), dh = 24 (row)
                                <1>       ; 18/11/2020
000022B3 B618                   <1>       mov    dh, 25-1
000022B5 8A15[BC6A0000]         <1>       mov    dl, [CRT_COLS]
000022BB FECA                   <1>       dec    dl
                                <1>       ;
000022BD B001                   <1>       mov    al, 1        ; scroll 1 line up
                                <1>              ; ah = attribute
                                <1>       ;mov   bl, al ; 12/05/2016
000022BF E9F0FCFFFF             <1>       jmp    _scroll_up   ; 16/01/2016
                                <1> ;u4:
                                <1>       ;;int  10h          ; video-call return
                                <1>                           ; scroll up the screen
                                <1>                           ; tty return
                                <1> ;u5:
                                <1>       ;retn               ; return to the caller
                                <1>
                                <1> u6:                       ; set-cursor-inc
000022C4 FEC6                   <1>       inc    dh           ; next row
```

```
                        <1>                             ; set cursor
                        <1> ;u7:
                        <1>         ;;mov  ah, 02h
                        <1>         ;;jmp  short u4     ; establish the new cursor
                        <1>         ;call  _set_cpos
                        <1>         ;jmp   short u5
000022C6 EB32           <1>         jmp   _set_cpos
                        <1>
                        <1>         ; check for control characters
                        <1> u8:
                        <1>         ;je    short u9 ; 17/04/2021
000022C8 3C0A           <1>         cmp    al, 0Ah              ; is it a line feed (0Ah)
000022CA 74D4           <1>         je     short u10
000022CC 3C07           <1>         cmp    al, 07h      ; is it a bell
000022CE 747C           <1>         je     short u11
000022D0 3C08           <1>         cmp    al, 08h              ; is it a backspace
                        <1>         ;jne   short u0
000022D2 741C           <1>         je     short bs     ; 12/12/2013
                        <1>         ; 12/12/2013 (tab stop)
000022D4 3C09           <1>         cmp    al, 09h              ; is it a tab stop
000022D6 75B3           <1>         jne    short u0
000022D8 88D0           <1>         mov    al, dl
                        <1>         ;cbw
000022DA 30E4           <1>         xor    ah, ah ; 09/12/2017
000022DC B108           <1>         mov    cl, 8
000022DE F6F1           <1>         div    cl
000022E0 28E1           <1>         sub    cl, ah
                        <1> ts:
                        <1>         ; 02/09/2014
                        <1>         ; 01/09/2014
                        <1>         ;mov   al, 20h
                        <1> tsloop:
                        <1>         ;push  cx
                        <1>         ; 12/04/2021
000022E2 51             <1>         push   ecx ; *
                        <1>         ; 18/11/2020
                        <1>         ;push  ax
                        <1>         ;;mov  bh, [ACTIVE_PAGE]
                        <1>         ; 05/12/2020
                        <1>         ;push  bx
000022E3 B020           <1>         mov    al, 20h ; al = space (blank)
                        <1>                        ; bl = color/attribute
000022E5 E88CFFFFFF     <1>         call   _write_tty_m3 ; 24/06/2016 (m3 -> _write_tty_m3)
                        <1>         ; 05/12/2020
                        <1>         ; bx is preserved in '_write_tty_m3'
                        <1>         ; 18/11/2020
                        <1>         ;pop   bx  ; BL = color/attribute, Bh = video page
                        <1>         ;pop   ax  ; AL = character
                        <1>         ; 12/04/2021
                        <1>         ;pop   cx
000022EA 59             <1>         pop    ecx ; *
000022EB FEC9           <1>         dec    cl
000022ED 75F3           <1>         jnz    short tsloop
000022EF C3             <1>         retn
                        <1> bs:
                        <1>         ; back space found
                        <1>
000022F0 08D2           <1>         or     dl, dl       ; is it already at start of line
                        <1>         ;je    short u7     ; set_cursor
000022F2 7406           <1>         jz     short _set_cpos
                        <1>         ;dec   dx           ; no -- just move it back
                        <1>         ; 17/04/2021
000022F4 FECA           <1>         dec    dl           ; move to 1 column back
                        <1>         ;jmp   short u7
000022F6 EB02           <1>         jmp    short _set_cpos
                        <1>
                        <1>         ; carriage return found
                        <1> u9:
000022F8 B200           <1>         mov    dl, 0        ; move to first column
                        <1>         ;jmp   short u7
                        <1>         ;jmp   short _set_cpos ; 30/01/2016
                        <1>
                        <1>         ; line feed found
                        <1> ;u10:
                        <1> ;       cmp    dh, 25-1     ; bottom of screen
                        <1> ;       jne    short u6     ; no, just set the cursor
                        <1> ;       jmp    u1            ; yes, scroll the screen
                        <1>
                        <1> _set_cpos:
                        <1>         ; 17/04/2021 - TRDOS 386 v2.0.4
                        <1>         ; 12/05/2016 - TRDOS 386 (TRDOS v2.0)
                        <1>         ; 27/06/2015
                        <1>         ; 01/09/2014
                        <1>         ; 30/08/2014 (Retro UNIX 386 v1)
                        <1>         ;
                        <1>         ; 04/12/2013 - 12/12/2013 (Retro UNIX 8086 v1)
                        <1>         ;
                        <1>         ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                        <1>         ;
                        <1> ;--------------------------------------------
                        <1> ; SET_CPOS
                        <1> ;       THIS ROUTINE SETS THE CURRENT CURSOR POSITION TO THE
                        <1> ;       NEW X-Y VALUES PASSED
                        <1> ; INPUT
                        <1> ;       DX - ROW,COLUMN OF NEW CURSOR
                        <1> ;       BH - DISPLAY PAGE OF CURSOR
                        <1> ; OUTPUT
                        <1> ;       CURSOR ID SET AT 6845 IF DISPLAY PAGE IS CURRENT DISPLAY
                        <1> ;--------------------------------------------
                        <1>         ;
000022FA BE[C6800100]   <1>         mov    esi, CURSOR_POSN
000022FF 0FB6C7         <1>         movzx eax, bh ; BH = video page number
                        <1> ;       or     al, al
                        <1> ;       jz     short _set_cpos_0
00002302 D0E0           <1>         shl    al, 1    ; word offset
```

```
6253 00002304 01C6                <1>       add   esi, eax
6254                              <1> ;_set_cpos_0:
6255 00002306 668916              <1>       mov   [esi], dx ; save the pointer
6256 00002309 383D[D6800100]      <1>       cmp   [ACTIVE_PAGE], bh
6257 0000230F 7532                <1>       jne   short m17
6258                              <1>       ;call  m18    ; CURSOR SET
6259                              <1> ;m17:               ; SET_CPOS_RETURN
6260                              <1>       ; 01/09/2014
6261                              <1> ;     retn
6262                              <1>             ; DX  = row/column
6263                              <1> m18:
6264 00002311 E849FCFFFF          <1>       call  position ; determine location in regen buffer
6265 00002316 668B0D[C4800100]    <1>       mov   cx, [CRT_START]
6266 0000231D 6601C1              <1>       add   cx, ax  ; add char position in regen buffer
6267                              <1>                     ; to the start address (offset) for this page
6268 00002320 66D1E9              <1>       shr   cx, 1  ; divide by 2 for char only count
6269 00002323 B40E                <1>       mov   ah, 14 ; register number for cursor
6270                              <1>       ;call  m16    ; output value to the 6845
6271                              <1>       ;retn
6272                              <1>
6273                              <1>       ;-----THIS ROUTINE OUTPUTS THE CX REGISTER
6274                              <1>       ;     TO THE 6845 REGISTERS NAMED IN (AH)
6275                              <1> m16:
6276 00002325 FA                  <1>       cli
6277                              <1>       ;mov   dx, [addr_6845] ; address register
6278 00002326 66BAD403            <1>       mov   dx, 03D4h ; I/O address of color card
6279 0000232A 88E0                <1>       mov   al, ah ; get value
6280 0000232C EE                  <1>       out   dx, al ; register set
6281                              <1>       ;inc   dx     ; data register
6282                              <1>       ; 17/04/2021
6283 0000232D FEC2                <1>       inc   dl
6284 0000232F EB00                <1>       jmp   $+2    ; i/o delay
6285 00002331 88E8                <1>       mov   al, ch ; data
6286 00002333 EE                  <1>       out   dx, al
6287                              <1>       ;dec   dx
6288                              <1>       ; 17/04/2021
6289 00002334 FECA                <1>       dec   dl
6290 00002336 88E0                <1>       mov   al, ah
6291 00002338 FEC0                <1>       inc   al     ; point to other data register
6292 0000233A EE                  <1>       out   dx, al ; set for second register
6293                              <1>       ;inc   dx
6294                              <1>       ; 17/04/2021
6295 0000233B FEC2                <1>       inc   dl
6296 0000233D EB00                <1>       jmp   $+2    ; i/o delay
6297 0000233F 88C8                <1>       mov   al, cl ; second data value
6298 00002341 EE                  <1>       out   dx, al
6299 00002342 FB                  <1>       sti
6300                              <1> m17:
6301 00002343 C3                  <1>       retn
6302                              <1>
6303                              <1> _beep:
6304                              <1>       ; 12/02/2021 (TRDOS v2.0.3)
6305 00002344 FA                  <1>       cli
6306 00002345 E811000000          <1>       call  beep
6307 0000234A FB                  <1>       sti
6308 0000234B C3                  <1>       retn
6309                              <1>
6310                              <1> beeper:
6311                              <1>       ; 04/08/2016
6312                              <1>       ; 12/05/2016 - TRDOS 386 (TRDOS v2.0)
6313                              <1>       ; 30/08/2014 (Retro UNIX 386 v1)
6314                              <1>       ; 18/01/2014
6315                              <1>       ; 03/12/2013
6316                              <1>       ; bell found
6317                              <1> u11:
6318 0000234C FB                  <1>       sti
6319 0000234D 3A3D[D6800100]      <1>       cmp   bh, [ACTIVE_PAGE]
6320 00002353 7552                <1>       jne   short u12    ; Do not sound the beep
6321                              <1>                          ; if it is not written on the active page
6322                              <1> beeper_gfx: ; 04/08/2016
6323 00002355 66B93305            <1>       mov   cx, 1331     ; divisor for 896 hz tone
6324 00002359 B31F                <1>       mov   bl, 31       ; set count for 31/64 second for beep
6325                              <1>       ;call  beep        ; sound the pod bell
6326                              <1>       ;jmp   short u5    ; tty_return
6327                              <1>       ;retn
6328                              <1>
6329                              <1> TIMER equ   040h         ; 8254 TIMER - BASE ADDRESS
6330                              <1> PORT_B equ  061h         ; PORT B READ/WRITE DIAGNOSTIC REGISTER
6331                              <1> GATE2 equ   00000001b    ; TIMER 2 INPUT CATE CLOCK BIT
6332                              <1> SPK2  equ   00000010b    ; SPEAKER OUTPUT DATA ENABLE BIT
6333                              <1>
6334                              <1> beep:
6335                              <1>       ; 12/02/2021
6336                              <1>       ; 07/02/2015
6337                              <1>       ; 30/08/2014 (Retro UNIX 386 v1)
6338                              <1>       ; 18/01/2014
6339                              <1>       ; 03/12/2013
6340                              <1>       ;
6341                              <1>       ; TEST4.ASM - 06/10/85  POST AND BIOS UTILITY ROUTINES
6342                              <1>       ;
6343                              <1>       ; ROUTINE TO SOUND THE BEEPER USING TIMER 2 FOR TONE
6344                              <1>       ;
6345                              <1>       ; ENTRY:
6346                              <1>       ;    (BL) = DURATION COUNTER ( 1 FOR 1/64 SECOND )
6347                              <1>       ;    (CX) = FREQUENCY DIVISOR (1193180/FREQUENCY) (1331 FOR 886 HZ)
6348                              <1>       ; EXIT:                          :
6349                              <1>       ;    (AX),(BL),(CX) MODIFIED.
6350                              <1>
6351 0000235B 9C                  <1>       pushfd  ; 18/01/2014      ; save interrupt status
6352 0000235C FA                  <1>       cli                 ; block interrupts during update
6353 0000235D B0B6                <1>       mov   al, 10110110b; select timer 2, lsb, msb binary
6354 0000235F E643                <1>       out   TIMER+3, al  ; write timer mode register
6355 00002361 EB00                <1>       jmp   $+2          ; I/O delay
6356 00002363 88C8                <1>       mov   al, cl       ; divisor for hz (low)
6357 00002365 E642                <1>       out   TIMER+2,AL   ; write timer 2 count - lsb
```

```
6358 00002367 EB00              <1>       jmp    $+2           ; I/O delay
6359 00002369 88E8              <1>       mov    al, ch        ; divisor for hz (high)
6360 0000236B E642              <1>       out    TIMER+2, al   ; write timer 2 count - msb
6361 0000236D E461              <1>       in     al, PORT_B    ; get current setting of port
6362 0000236F 88C4              <1>       mov    ah, al        ; save that setting
6363 00002371 0C03              <1>       or     al, GATE2+SPK2      ; gate timer 2 and turn speaker on
6364 00002373 E661              <1>       out    PORT_B, al    ; and restore interrupt status
6365                            <1>       ; 12/02/2021
6366 00002375 9D                <1>       popfd  ; 18/01/2014
6367                            <1>       ;sti
6368                            <1> g7:                         ; 1/64 second per count (bl)
6369 00002376 B90B040000        <1>       mov    ecx, 1035      ; delay count for 1/64 of a second
6370 0000237B E828000000        <1>       call   waitf          ; go to beep delay 1/64 count
6371 00002380 FECB              <1>       dec    bl             ; (bl) length count expired?
6372 00002382 75F2              <1>       jnz    short g7       ; no - continue beeping speaker
6373                            <1>       ;
6374 00002384 9C                <1>       pushfd ; 12/02/2021 ; save interrupt status
6375 00002385 FA                <1>       cli    ; 18/01/2014 ; block interrupts during update
6376 00002386 E461              <1>       in     al, PORT_B    ; get current port value
6377                            <1>       ;or   al, not (GATE2+SPK2) ; isolate current speaker bits in case
6378 00002388 0CFC              <1>       or    al, ~(GATE2+SPK2)
6379 0000238A 20C4              <1>       and    ah, al        ; someone turned them off during beep
6380 0000238C 88E0              <1>       mov    al,ah         ; recover value of port
6381                            <1>      ;or    al, not (GATE2+SPK2) ; force speaker data off
6382 0000238E 0CFC              <1>       or     al, ~(GATE2+SPK2) ; isolate current speaker bits in case
6383 00002390 E661              <1>       out    PORT_B, al    ; and stop speaker timer
6384 00002392 9D                <1>       popfd  ; 12/02/2021 ; restore interrupt flag state
6385                            <1>       ;sti
6386                            <1>       ;mov   ecx, 1035     ; force 1/64 second delay (short)
6387                            <1>       ; 17/04/2021
6388 00002393 66B90B04          <1>       mov    cx, 1035
6389 00002397 E80C000000        <1>       call   waitf          ; minimum delay between all beeps
6390 0000239C 9C                <1>       pushfd                ; save interrupt status
6391 0000239D FA                <1>       cli                   ; block interrupts during update
6392 0000239E E461              <1>       in     al, PORT_B    ; get current port value in case
6393 000023A0 2403              <1>       and    al, GATE2+SPK2 ; someone turned them on
6394 000023A2 08E0              <1>       or     al, ah        ; recover value of port_b
6395 000023A4 E661              <1>       out    PORT_B, al    ; restore speaker status
6396 000023A6 9D                <1>       popfd                ; restore interrupt flag state
6397                            <1> u12:
6398 000023A7 C3                <1>       retn
6399                            <1>
6400                            <1> REFRESH_BIT equ    00010000b    ; REFRESH TEST BIT
6401                            <1>
6402                            <1> WAITF:
6403                            <1> waitf:
6404                            <1>       ; 30/08/2014 (Retro UNIX 386 v1)
6405                            <1>       ; 03/12/2013
6406                            <1>       ;
6407                            <1> ;     push ax             ; save work register (ah)
6408                            <1> ;waitf1:
6409                            <1>                           ; use timer 1 output bits
6410                            <1> ;     in     al, PORT_B   ; read current counter output status
6411                            <1> ;     and    al, REFRESH_BIT     ; mask for refresh determine bit
6412                            <1> ;     cmp    al, ah       ; did it just change
6413                            <1> ;     je     short waitf1 ; wait for a change in output line
6414                            <1> ;     ;
6415                            <1> ;     mov    ah, al       ; save new lflag state
6416                            <1> ;     loop   waitf1       ; decrement half cycles till count end
6417                            <1> ;     ;
6418                            <1> ;     pop    ax           ; restore (ah)
6419                            <1> ;     retn                ; return (cx)=0
6420                            <1>
6421                            <1> ; 06/02/2015 (unix386.s <-- dsectrm2.s)
6422                            <1> ; 17/12/2014 (dsectrm2.s)
6423                            <1> ; WAITF
6424                            <1> ; /// IBM PC-XT Model 286 System BIOS Source Code - Test 4 - 06/10/85 ///
6425                            <1> ;
6426                            <1> ;---WAITF-----------------------------------------------------------------------
6427                            <1> ;     FIXED TIME WAIT ROUTINE (HARDWARE CONTROLLED - NOT PROCESSOR)
6428                            <1> ; ENTRY:
6429                            <1> ;     (CX) = COUNT OF 15.085737 MICROSECOND INTERVALS TO WAIT
6430                            <1> ;           MEMORY REFRESH TIMER 1 OUTPUT USED AS REFERENCE
6431                            <1> ; EXIT:
6432                            <1> ;                 AFTER (CX) TIME COUNT (PLUS OR MINUS 16 MICROSECONDS)
6433                            <1> ;     (CX) = 0
6434                            <1> ;-------------------------------------------------------------------------------
6435                            <1>
6436                            <1> ; Refresh period: 30 micro seconds (15-80 us)
6437                            <1> ; (16/12/2014 - AWARDBIOS 1999 - ATORGS.ASM, WAIT_REFRESH)
6438                            <1>
6439                            <1> ;WAITF:                        ; DELAY FOR (CX)*15.085737 US
6440                            <1>       ;PUSH  AX               ; SAVE WORK REGISTER (AH)
6441                            <1>       ; 11/04/2021
6442 000023A8 50                <1>       push   eax
6443                            <1>       ; 16/12/2014
6444                            <1>       ;shr   cx, 1            ; convert to count of 30 micro seconds
6445 000023A9 D1E9              <1>       shr    ecx, 1 ; 21/02/2015
6446                            <1> ;17/12/2014
6447                            <1> ;WAITF1:
6448                            <1> ;     IN     AL, PORT_B   ;061h  ; READ CURRENT COUNTER OUTPUT STATUS
6449                            <1> ;     AND    AL, REFRESH_BIT     ;00010000b ; MASK FOR REFRESH DETERMINE BIT
6450                            <1> ;     CMP    AL, AH              ; DID IT JUST CHANGE
6451                            <1> ;     JE     short WAITF1        ; WAIT FOR A CHANGE IN OUTPUT LINE
6452                            <1> ;     MOV    AH, AL              ; SAVE NEW FLAG STATE
6453                            <1> ;     LOOP   WAITF1              ; DECREMENT HALF CYCLES TILL COUNT END
6454                            <1>       ;
6455                            <1>       ; 17/12/2014
6456                            <1>       ;
6457                            <1>       ; Modification from 'WAIT_REFRESH' procedure of AWARD BIOS - 1999
6458                            <1>       ;
6459                            <1> ;WAIT_REFRESH:  Uses port 61, bit 4 to have CPU speed independent waiting.
6460                            <1> ;     INPUT:  CX = number of refresh periods to wait
6461                            <1> ;                  (refresh periods = 1 per 30 microseconds on most machines)
6462                            <1> WR_STATE_0:
```

```
6463 000023AB E461              <1>       IN     AL,PORT_B          ; IN AL,SYS1
6464 000023AD A810              <1>       TEST   AL,010H
6465 000023AF 74FA              <1>       JZ     SHORT WR_STATE_0
6466                            <1> WR_STATE_1:
6467 000023B1 E461              <1>       IN     AL,PORT_B          ; IN AL,SYS1
6468 000023B3 A810              <1>       TEST   AL,010H
6469 000023B5 75FA              <1>       JNZ    SHORT WR_STATE_1
6470 000023B7 E2F2              <1>         LOOP    WR_STATE_0
6471                            <1>       ;
6472                            <1>       ;POP   AX                 ; RESTORE (AH)
6473                            <1>       ; 11/04/2021
6474 000023B9 58                <1>       pop    eax
6475 000023BA C3                <1>       RETn                      ; (CX) = 0
6476                            <1>
6477                            <1> ; 09/07/2016
6478                            <1> ; 01/07/2016
6479                            <1> ; 24/06/2016
6480                            <1> ; 23/06/2016 - TRDOS 386 (TRDOS v2.0)
6481                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
6482                            <1> ;--------------------------------------------------------------------------------
6483                            <1> ; WRITE_STRING                                                              :
6484                            <1> ;     THIS ROUTINE WRITES A STRING OF CHARACTERS TO THE CRT.                 :
6485                            <1> ; INPUT                                                                     :
6486                            <1> ;     (AL) = WRITE STRING COMMAND  0 - 3                                    :
6487                            <1> ;     (BH) = DISPLAY PAGE (ACTIVE PAGE)                                     :
6488                            <1> ;     (CX) = COUNT OF CHARACTERS TO WRITE, IF (CX) = 0 THEN RETURN          :
6489                            <1> ;     (DX) = CURSOR POSITION FOR START OF STRING WRITE                       :
6490                            <1> ;     (BL) = ATTRIBUTE OF CHARACTER TO WRITE IF (AL) = 0  OR     (AL) = 1   :
6491                            <1> ;     (eBP) = SOURCE STRING OFFSET                                          :
6492                            <1> ; OUTPUT                                                                    :
6493                            <1> ;     NONE                                                                  :
6494                            <1> ;--------------------------------------------------------------------------------
6495                            <1>
6496                            <1> ; AL = 00h: Assign all characters the attribute in BL; do not update cursor
6497                            <1> ; AL = 01h: Assign all characters the attribute in BL; update cursor
6498                            <1> ; AL = 02h: Use attributes in string; do not update cursor
6499                            <1> ; AL = 03h: Use attributes in string; update cursor
6500                            <1>
6501                            <1> WRITE_STRING:
6502                            <1>       ; 12/09/2016
6503                            <1>       ; 09/07/2016
6504                            <1>       ;cmp   byte [CRT_MODE], 7 ; 6?!
6505                            <1>       ;ja    VIDEO_RETURN       ; not a valid function for VGA modes
6506                            <1>       ;
6507 000023BB A2[3C8D0100]     <1>       mov    [w_str_cmd], al    ; save (AL) command
6508 000023C0 3C04             <1>       CMP    AL, 4              ; TEST FOR INVALID WRITE STRING OPTION
6509 000023C2 0F8321F7FFFF     <1>       JNB    VIDEO_RETURN       ; IF OPTION INVALID THEN RETURN
6510                            <1>
6511                            <1>      ;JCXZ  VIDEO_RETURN       ; IF ZERO LENGTH STRING THEN RETURN
6512                            <1>
6513 000023C8 67E362           <1>       jcxz   P55                ; 01/07/2016
6514                            <1>
6515                            <1>       ; 01/07/2016
6516                            <1>       ;and   ecx, 0FFFFh
6517                            <1>       ; ECX = byte count
6518                            <1>       ;push  ecx
6519 000023CB 89EE             <1>       mov    esi, ebp ; user buffer
6520 000023CD BF00000700       <1>       mov    edi, Cluster_Buffer  ; system buffer
6521 000023D2 E8B5F10000       <1>       call   transfer_from_user_buffer
6522                            <1>       ;pop   ecx
6523 000023D7 0F820CF7FFFF     <1>       jc     VIDEO_RETURN
6524                            <1>       ; ecx = transfer (byte) count = character count
6525 000023DD BD00000700       <1>       mov    ebp, Cluster_Buffer
6526                            <1>       ; 12/09/2016
6527 000023E2 803D[BA6A0000]07 <1>       cmp    byte [CRT_MODE], 7 ; 6?!
6528 000023E9 0F87A1000000     <1>       ja     vga_write_string
6529                            <1>       ;
6530 000023EF 0FB6F7           <1>       movzx  esi, bh            ; GET CURRENT CURSOR PAGE
6531 000023F2 66D1E6           <1>       SAL    SI, 1              ; CONVERT TO PAGE OFFSET  (SI= PAGE)
6532                            <1>       ; *****
6533 000023F5 66FFB6[C6800100] <1>       PUSH   word [eSI+CURSOR_POSN] ; SAVE CURRENT CURSOR POSITION IN STACK
6534                            <1>
6535                            <1>       ;MOV   AX,0200H           ; SET NEW CURSOR POSITION
6536                            <1>       ;INT   10H
6537                            <1> P50next:
6538 000023FC 51               <1>       push   ecx ; ****
6539 000023FD 53               <1>       push   ebx ; *** ; 18/11/2020
6540 000023FE 56               <1>       push   esi ; **
6541 000023FF 52               <1>       push   edx ; *
6542 00002400 E8F5FEFFFF       <1>       call   _set_cpos
6543                            <1> P50:
6544 00002405 8A4500           <1>       MOV    AL, [eBP]          ; GET CHARACTER FROM INPUT STRING
6545 00002408 45               <1>       INC    eBP                ; BUMP POINTER TO CHARACTER
6546                            <1>
6547                            <1> ;-----      TEST FOR SPECIAL CHARACTER'S
6548                            <1>
6549 00002409 3C08             <1>       CMP    AL, 08H            ; IS IT A BACKSPACE
6550 0000240B 7410             <1>       JE     short P51          ; BACK_SPACE
6551 0000240D 3C0D             <1>       CMP    AL, 0Dh ; CR       ; IS IT CARRIAGE RETURN
6552 0000240F 740C             <1>       JE     short P51          ; CAR_RET
6553 00002411 3C0A             <1>       CMP    AL, 0Ah ; LF       ; IS IT A LINE FEED
6554 00002413 7408             <1>       JE     short P51          ; LINE_FEED
6555                            <1>       ; 18/11/2020
6556 00002415 3C09             <1>       cmp    al, 09h            ; is it a tab stop
6557 00002417 7404             <1>       je     short P51
6558                            <1>       ;
6559 00002419 3C07             <1>       CMP    AL, 07h            ; IS IT A BELL
6560 0000241B 7515             <1>       JNE    short P52          ; IF NOT THEN DO WRITE CHARACTER
6561                            <1> P51:
6562                            <1>       ;MOV   AH,0EH             ; TTY_CHARACTER_WRITE
6563                            <1>       ;INT   10H                ; WRITE TTY CHARACTER TO THE CRT
6564                            <1>
6565 0000241D E854FEFFFF       <1>       call   _write_tty_m3
6566                            <1>
6567 00002422 5A               <1>       pop    edx ; *
```

```
6568 00002423 5E                  <1>       pop    esi ; **
6569                              <1>
6570 00002424 668B96[C6800100]    <1>       MOV    DX, [eSI+CURSOR_POSN] ; GET CURRENT CURSOR POSITION
6571 0000242B EB44                <1>       JMP    SHORT P54           ; SET CURSOR POSITION AND CONTINUE
6572                              <1> P55:
6573 0000242D E9B7F6FFFF          <1>       JMP    VIDEO_RETURN
6574                              <1> P52:
6575 00002432 66B90100            <1>       MOV    CX, 1               ; SET CHARACTER WRITE AMOUNT TO ONE
6576 00002436 803D[3C8D0100]02    <1>       CMP    byte [w_str_cmd], 2 ; IS THE ATTRIBUTE IN THE STRING
6577 0000243D 7204                <1>       JB     short P53           ; IF NOT THEN SKIP
6578 0000243F 8A5D00              <1>       MOV    BL, [eBP]           ; ELSE GET NEW ATTRIBUTE
6579 00002442 45                  <1>       INC    eBP                 ; BUMP STRING POINTER
6580                              <1> P53:
6581                              <1>       ;MOV   AH,09H              ; GOT_CHARACTER
6582                              <1>       ;INT   10H                 ; WRITE CHARACTER TO THE CRT
6583                              <1>
6584 00002443 E89AFDFFFF          <1>       call   _write_c_current
6585                              <1>
6586 00002448 5A                  <1>       pop    edx ; *
6587                              <1>
6588                              <1>       ; 05/12/2020
6589                              <1>       ; bx is preserved in '_write_c_current'
6590                              <1>       ; 18/11/2020
6591                              <1>       ;mov   ebx, [esp+4] ; ***
6592                              <1>
6593 00002449 0FB6F7              <1>       movzx  esi, bh ; video page number (0 to 7)
6594 0000244C 889E[C36A0000]      <1>       mov    [esi+chr_attrib], bl ; color/attribute
6595                              <1>
6596 00002452 FEC2                <1>       INC    DL                  ; INCREMENT COLUMN COUNTER
6597 00002454 3A15[BC6A0000]      <1>       CMP    DL, [CRT_COLS]      ; IF COLS ARE WITHIN RANGE FOR THIS MODE
6598                              <1>       ;JB    short P54           ;  THEN GO TO COLUMNS SET
6599 0000245A 7214                <1>       jb     short P56 ; 05/12/2020
6600 0000245C FEC6                <1>       INC    DH                  ; BUMP ROW COUNTER BY ONE
6601 0000245E 28D2                <1>       SUB    DL, DL              ; SET COLUMN COUNTER TO ZERO
6602 00002460 80FE19              <1>       CMP    DH, 25              ; IF ROWS ARE LESS THAN 25 THEN
6603                              <1>       ;JB    short P54           ; GO TO ROWS_COLUMNS_SET
6604 00002463 720B                <1>       jb     short P56 ; 05/12/2020
6605                              <1>
6606                              <1>       ; 18/11/2020
6607                              <1>       ;MOV   AX,0E0AH            ; ELSE SCROLL SCREEN
6608                              <1>       ;INT   10H                 ; RESET ROW COUNTER TO 24
6609                              <1>
6610                              <1>       ; 18/11/2020
6611 00002465 B00A                <1>       mov    al, 0Ah      ; line feed
6612                              <1>
6613 00002467 E80AFEFFFF          <1>       call   _write_tty_m3
6614                              <1>
6615 0000246C 66BA0018            <1>       mov    dx, 1800h           ; Column = 0, Row = 24
6616                              <1> P56:
6617                              <1>       ; 05/12/2020
6618                              <1>       ; 18/11/2020
6619 00002470 5E                  <1>       pop    esi ; **
6620                              <1> P54:                              ; ROW_COLUMNS_SET
6621                              <1>       ;MOV   AX,0200H            ; SET NEW CURSOR POSITION COMMAND
6622                              <1>       ;INT   10H                 ; ESTABLISH NEW CURSOR POSITION
6623                              <1>
6624                              <1>       ; 18/11/2020
6625 00002471 5B                  <1>       pop    ebx ; ***
6626 00002472 59                  <1>       pop    ecx ; ****
6627                              <1>
6628                              <1>       ;LOOP  P50                 ; DO IT ONCE MORE UNTIL (CX) = ZERO
6629 00002473 6649                <1>       dec    cx
6630 00002475 7585                <1>       jnz    short P50next
6631                              <1>
6632 00002477 665A                <1>       POP    DX  ; *****         ; RESTORE OLD CURSOR COORDINATES
6633                              <1>
6634 00002479 F605[3C8D0100]01    <1>       test   byte [w_str_cmd], 1 ; IF CURSOR WAS NOT TO BE MOVED
6635 00002480 0F8563F6FFFF        <1>       JNZ    VIDEO_RETURN        ; THEN EXIT WITHOUT RESETTING OLD VALUE
6636                              <1>
6637                              <1>       ;MOV   AX,0200H            ; ELSE RESTORE OLD CURSOR POSITION
6638                              <1>       ;INT   10H
6639                              <1>                                   ; DONE - EXIT WRITE STRING
6640 00002486 E86FFEFFFF          <1>       call   _set_cpos
6641 0000248B E959F6FFFF          <1>       JMP    VIDEO_RETURN        ; RETURN TO CALLER
6642                              <1>
6643                              <1> vga_write_string:
6644                              <1>       ; 12/09/2016 - TRDOS 386 (TRDOS v2.0)
6645                              <1>       ;
6646                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
6647                              <1>       ; vgabios-0.7a (2011)
6648                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
6649                              <1>       ; 'vgabios.c', ' biosfn_write_string'
6650                              <1>
6651                              <1>       ; INPUT                                                                 :
6652                              <1>       ;      (AL) = WRITE STRING COMMAND  0 - 3                               :
6653                              <1>       ;      (BH) = DISPLAY PAGE (ACTIVE PAGE)                                :
6654                              <1>       ;      (CX) = COUNT OF CHARACTERS TO WRITE, IF (CX) = 0 THEN RETURN     :
6655                              <1>       ;      (DX) = CURSOR POSITION FOR START OF STRING WRITE                  :
6656                              <1>       ;      (BL) = ATTRIBUTE OF CHARACTER TO WRITE IF (AL) = 0  OR (AL) = 1   :
6657                              <1>       ;      (eBP) = SOURCE STRING OFFSET                                     :
6658                              <1>       ; OUTPUT                                                                :
6659                              <1>       ;      NONE                                                             :
6660                              <1>       ;-------------------------------------------------------------------------
6661                              <1>
6662                              <1>       ; AL = 00h: Assign all characters the attribute in BL; do not update cursor
6663                              <1>       ; AL = 01h: Assign all characters the attribute in BL; update cursor
6664                              <1>       ; AL = 02h: Use attributes in string; do not update cursor
6665                              <1>       ; AL = 03h: Use attributes in string; update cursor
6666                              <1>
6667                              <1>       ; biosfn_write_string(GET_AL(),GET_BH(),GET_BL(),CX,GET_DH(),GET_DL(),ES,BP);
6668                              <1>       ; static void biosfn_write_string (flag,page,attr,count,row,col,seg,offset)
6669                              <1>
6670                              <1>       ; // Read curs info for the page
6671                              <1>       ; biosfn_get_cursor_pos(page,&dummy,&oldcurs);
6672                              <1>       ; bh = video page = 0
```

```
6673                                 <1>       ;movzx esi, word [CURSOR_POSN] ; current cursor position for video page 0
6674                                 <1>
6675                                 <1>       ; // if row=0xff special case : use current cursor position
6676                                 <1>       ; if(row==0xff)
6677                                 <1>       ;  {col=oldcurs&0x00ff;
6678                                 <1>       ;   row=(oldcurs&0xff00)>>8;
6679                                 <1>       ;  }
6680                                 <1>
6681                                 <1>       ;mov   al, [w_str_cmd]
6682                                 <1>
6683 00002490 80FEFF                 <1>       cmp    dh, 0FFh
6684 00002493 7407                   <1>       je     short vga_wstr_1 ; user current cursor position
6685                                 <1> vga_wstr_0:
6686                                 <1>       ; set cursor position
6687 00002495 668915[C6800100]       <1>       mov    [CURSOR_POSN], dx ; save cursor pos for pg 0
6688                                 <1> vga_wstr_1:
6689 0000249C 66FF35[C6800100]       <1>       push   word [CURSOR_POSN] ; *
6690                                 <1>
6691                                 <1>       ; ebp = string offset in system buffer (user buffer was copied to)
6692                                 <1>
6693                                 <1>       ; while(count--!=0)
6694                                 <1>       ;  {
6695                                 <1>       ;   car=read_byte(seg,offset++);
6696                                 <1>       ;   if((flag&0x02)!=0)
6697                                 <1>       ;    attr=read_byte(seg,offset++);
6698                                 <1>       ;    biosfn_write_teletype(car,page,attr,WITH_ATTR);
6699                                 <1>       ;  }
6700                                 <1>
6701                                 <1>       ;push  eax ; **
6702                                 <1>       ;test  al, 2
6703 000024A3 F605[3C8D0100]02       <1>       test   byte [w_str_cmd], 2
6704 000024AA 751D                   <1>       jnz    short vga_wstr_3
6705 000024AC 881D[D7800100]         <1>       mov    [ccolor], bl
6706                                 <1> vga_wstr_2:
6707 000024B2 51                     <1>       push   ecx
6708 000024B3 8A4500                 <1>       mov    al, [ebp]
6709 000024B6 E8C90A0000             <1>       call   vga_write_teletype
6710 000024BB 59                     <1>       pop    ecx
6711 000024BC 6649                   <1>       dec    cx
6712 000024BE 741E                   <1>       jz     short vga_wstr_4
6713 000024C0 45                     <1>       inc    ebp
6714 000024C1 8A1D[D7800100]         <1>       mov    bl, [ccolor]
6715 000024C7 EBE9                   <1>       jmp    short vga_wstr_2
6716                                 <1> vga_wstr_3:
6717 000024C9 51                     <1>       push   ecx
6718 000024CA 8A4500                 <1>       mov    al, [ebp]
6719 000024CD 45                     <1>       inc    ebp
6720 000024CE 8A5D00                 <1>       mov    bl, [ebp]
6721 000024D1 E8AE0A0000             <1>       call   vga_write_teletype
6722 000024D6 59                     <1>       pop    ecx
6723 000024D7 6649                   <1>       dec    cx
6724 000024D9 7403                   <1>       jz     short vga_wstr_4
6725 000024DB 45                     <1>       inc    ebp
6726 000024DC EBEB                   <1>       jmp    short vga_wstr_3
6727                                 <1> vga_wstr_4:
6728                                 <1>       ; // Set back curs pos
6729                                 <1>       ; if((flag&0x01)==0)
6730                                 <1>       ;  biosfn_set_cursor_pos(page,oldcurs);
6731                                 <1>       ; }
6732                                 <1>       ;pop   eax ; **
6733 000024DE 665A                   <1>       pop    dx ; word [CURSOR_POSN] ; *
6734                                 <1>       ;test  al, 1
6735 000024E0 F605[3C8D0100]01       <1>       test   byte [w_str_cmd], 1
6736 000024E7 0F85FCF5FFFF           <1>       jnz    VIDEO_RETURN
6737 000024ED 668915[C6800100]       <1>       mov    [CURSOR_POSN], dx
6738 000024F4 E9F0F5FFFF             <1>       JMP    VIDEO_RETURN
6739                                 <1>
6740                                 <1> ; 07/07/2016
6741                                 <1> ; 27/06/2016 - TRDOS 386 (TRDOS v2.0)
6742                                 <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
6743                                 <1> ;-----------------------------------------------------
6744                                 <1> ;  SCROLL UP
6745                                 <1> ;   THIS ROUTINE SCROLLS UP THE INFORMATION ON THE CRT
6746                                 <1> ; ENTRY ---
6747                                 <1> ;  CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL
6748                                 <1> ;  DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL
6749                                 <1> ;   BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS
6750                                 <1> ;  BH = FILL VALUE FOR BLANKED LINES
6751                                 <1> ;  AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE FIELD)
6752                                 <1> ;  DS = DATA SEGMENT
6753                                 <1> ;  ES = REGEN SEGMENT
6754                                 <1> ; EXIT --
6755                                 <1> ;  NOTHING, THE SCREEN IS SCROLLED
6756                                 <1> ;-----------------------------------------------------
6757                                 <1>
6758                                 <1>       ; cl = upper left column
6759                                 <1>       ; ch = upper left row
6760                                 <1>       ; dl = lower rigth column
6761                                 <1>       ; dh = lower right row
6762                                 <1>       ;
6763                                 <1>       ; al = line count (AL=0 means blank entire fields)
6764                                 <1>       ; bl = fill value for blanked lines
6765                                 <1>       ; bh = unused
6766                                 <1>
6767                                 <1> GRAPHICS_UP:
6768                                 <1>       ; 07/07/2016
6769                                 <1>       ;AH = Current video mode, [CRT_MODE]
6770 000024F9 80FC07                <1>       cmp    ah, 7
6771 000024FC 7766                   <1>       ja     short vga_graphics_up
6772                                 <1>       ;je    n0
6773                                 <1>
6774 000024FE 88C7                   <1>       MOV    bh, al              ; save line count in BH
6775 00002500 6689C8                 <1>       MOV    AX, CX              ; GET UPPER LEFT POSITION INTO AX REG
6776                                 <1>
6777                                 <1> ;-----      USE CHARACTER SUBROUTINE FOR POSITIONING
```

```
                                <1> ;-----      ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE
                                <1>
00002503 E8D0050000             <1>       CALL   GRAPH_POSN
00002508 0FB7F8                 <1>       MOVzx  eDI, AX             ; SAVE RESULT AS DESTINATION ADDRESS
                                <1>
                                <1> ;-----      DETERMINE SIZE OF WINDOW
                                <1>
0000250B 6629CA                 <1>       SUB    DX, CX
0000250E 6681C20101             <1>       ADD    DX, 101h            ; ADJUST VALUES
00002513 C0E602                 <1>       SAL    DH, 2               ; MULTIPLY ROWS BY 4 AT 8 VERT DOTS/CHAR
                                <1>                                  ; AND EVEN/ODD ROWS
                                <1> ;-----      DETERMINE CRT MODE
                                <1>
00002516 803D[BA6A0000]06       <1>       CMP    byte [CRT_MODE], 6  ; TEST FOR MEDIUM RES
0000251D 7305                   <1>       JNC    short _R7_              ; FIND_SOURCE
                                <1>
                                <1> ;-----      MEDIUM RES UP
0000251F D0E2                   <1>       SAL    DL, 1               ; # COLUMNS * 2, SINCE 2 BYTES/CHAR
00002521 66D1E7                 <1>       SAL    DI, 1               ; OFFSET *2 SINCE 2 BYTES/CHAR
                                <1>
                                <1> ;-----      DETERMINE THE SOURCE ADDRESS IN THE BUFFER
                                <1> _R7_:                           ; FIND_SOURCE
00002524 81C700800B00           <1>       add    edi, 0B8000h
0000252A C0E702                 <1>       sal    bh, 2               ; multiply number of lines by 4
0000252D 7431                   <1>       JZ     short _R11          ; IF ZERO, THEN BLANK ENTIRE FIELD
0000252F B050                   <1>       MOV    AL, 80              ; 80 BYTES/ROW
00002531 F6E7                   <1>       mul    bh                  ; determine offset to source
00002533 0FB7F0                 <1>       movzx  esi, ax                   ; offset to source
00002536 01FE                   <1>       add    eSI, eDI            ; SET UP SOURCE
00002538 88F4                   <1>       MOV    AH, DH              ; NUMBER OF ROWS IN FIELD
0000253A 28FC                   <1>       sub    ah, bh              ; determine number to move
                                <1>
                                <1> ;-----      LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS
                                <1> _R8:                            ; ROW_LOOP
0000253C E80B040000             <1>       CALL   _R17               ; MOVE ONE ROW
00002541 6681EEB01F             <1>       SUB    SI, 2000h-80       ; MOVE TO NEXT ROW
00002546 6681EFB01F             <1>       SUB    DI, 2000h-80
0000254B FECC                   <1>       DEC    AH                 ; NUMBER OF ROWS TO MOVE
0000254D 75ED                   <1>       JNZ    short _R8          ; CONTINUE TILL ALL MOVED
                                <1>
                                <1> ;-----      FILL IN THE VACATED LINE(S)
                                <1> _R9:                            ; CLEAR ENTRY
0000254F 88D8                   <1>       mov    al, bl             ; attribute to fill with
                                <1> _R10_:
00002551 E812040000             <1>       CALL   _R18               ; CLEAR THAT ROW
00002556 6681EFB01F             <1>       SUB    DI, 2000h-80       ; POINT TO NEXT LINE
0000255B FECF                   <1>       dec    bh                 ; number of lines to fill
0000255D 75F2                   <1>       JNZ    short _R10_        ; CLEAR LOOP
0000255F C3                     <1>       retn                      ; EVERYYHING DONE
                                <1>
                                <1> _R11:                           ; BLANK_FIELD
00002560 88F7                   <1>       mov    bh, dh             ; set blank count to everything in field
00002562 EBEB                   <1>       JMP    short _R9          ; CLEAR THE FIELD
                                <1>
                                <1> vga_graphics_up:
                                <1>       ; 12/04/2021
                                <1>       ; 08/08/2016
                                <1>       ; 07/08/2016
                                <1>       ; 04/08/2016
                                <1>       ; 01/08/2016
                                <1>       ; 31/07/2016
                                <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ;
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
                                <1>       ; 'vgabios.c', 'biosfn_scroll'
                                <1>       ;
                                <1>
                                <1>       ; cl = upper left column
                                <1>       ; ch = upper left row
                                <1>       ; dl = lower rigth column
                                <1>       ; dh = lower right row
                                <1>       ;
                                <1>       ; al = line count (AL=0 means blank entire fields)
                                <1>       ; bl = fill value for blanked lines
                                <1>       ; bh = unused
                                <1>       ;
                                <1>       ; ah = [CRT_MODE], current video mode
                                <1>
00002564 88C7                   <1>       mov    bh, al ; 31/07/2016
00002566 BE[DE6A0000]           <1>       mov    esi, vga_g_modes
0000256B 89F7                   <1>       mov    edi, esi
0000256D 83C708                 <1>       add    edi, vga_g_mode_count
                                <1> vga_g_up_0:
00002570 AC                     <1>       lodsb
00002571 38E0                   <1>       cmp    al, ah ; [CRT_MODE]
00002573 7405                   <1>       je     short vga_g_up_1
00002575 39FE                   <1>       cmp    esi, edi
00002577 72F7                   <1>       jb     short vga_g_up_0
                                <1>       ;xor   bh, bh ; 31/07/2016)
00002579 C3                     <1>       retn   ; nothing to do
                                <1> vga_g_up_1:
0000257A 88F8                   <1>       mov    al, bh ; 31/07/2016
0000257C 83C64F                 <1>       add    esi, vga_g_memmodel - (vga_g_modes + 1)
                                <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
                                <1>
                                <1>       ; if(rlr>=nbrows)rlr=nbrows-1;
                                <1>       ; if(clr>=nbcols)clr=nbcols-1;
                                <1>       ; if(nblines>nbrows)nblines=0;
                                <1>       ; cols=clr-cul+1;
                                <1>
0000257F 3A35[C26A0000]         <1>       cmp    dh, [VGA_ROWS]
00002585 7208                   <1>       jb     short vga_g_up_2
00002587 8A35[C26A0000]         <1>       mov    dh, [VGA_ROWS]
```

```
6883 0000258D FECE                <1>       dec    dh
6884                              <1> vga_g_up_2:
6885 0000258F 3A15[BC6A0000]      <1>       cmp    dl, [CRT_COLS]  ; = [VGA_COLS]
6886 00002595 7208                <1>       jb     short vga_g_up_3
6887 00002597 8A15[BC6A0000]      <1>       mov    dl, [CRT_COLS]
6888 0000259D FECA                <1>       dec    dl
6889                              <1> vga_g_up_3:
6890 0000259F 3A05[C26A0000]      <1>       cmp    al, [VGA_ROWS]
6891 000025A5 7602                <1>       jna    short vga_g_up_4
6892 000025A7 28C0                <1>       sub    al, al ; 0
6893                              <1> vga_g_up_4:
6894 000025A9 88D7                <1>       mov    bh, dl ; clr
6895 000025AB 28CF                <1>       sub    bh, cl ; cul
6896 000025AD FEC7                <1>       inc    bh ; cols = clr-cul+1
6897                              <1>
6898 000025AF 20C0                <1>       and    al, al ; nblines = 0
6899 000025B1 7559                <1>       jnz    short vga_g_up_6
6900 000025B3 20ED                <1>       and    ch, ch ; rul = 0
6901 000025B5 7555                <1>       jnz    short vga_g_up_6
6902 000025B7 20C9                <1>       and    cl, cl ; cul = 0
6903 000025B9 7551                <1>       jnz    short vga_g_up_6
6904                              <1>
6905                              <1>       ;push  ax
6906                              <1>       ; 12/04/2021
6907 000025BB 50                  <1>       push   eax
6908 000025BC A0[C26A0000]        <1>       mov    al, [VGA_ROWS]
6909 000025C1 FEC8                <1>       dec    al
6910 000025C3 38C6                <1>       cmp    dh, al ; rlr = nbrows-1
6911 000025C5 7545                <1>       jne    short vga_g_up_5
6912 000025C7 A0[BC6A0000]        <1>         mov     al, [CRT_COLS]  ; = VGA_COLS
6913 000025CC FEC8                <1>       dec    al
6914 000025CE 38C2                <1>       cmp    dl, al ; clr = nbcols-1
6915                              <1>       ;jne   short vga_g_up_5
6916                              <1>       ;;pop  ax
6917                              <1>       ; 12/04/2021
6918 000025D0 58                  <1>       pop    eax
6919 000025D1 7539                <1>       jne    short vga_g_up_5
6920                              <1>
6921 000025D3 66B80502            <1>       mov    ax, 0205h
6922 000025D7 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
6923 000025DB 66EF                <1>       out    dx, ax
6924 000025DD A0[C26A0000]        <1>       mov    al, [VGA_ROWS]
6925 000025E2 8A25[BC6A0000]      <1>       mov    ah, [CRT_COLS] ; = [VGA_COLS]
6926 000025E8 F6E4                <1>       mul    ah
6927 000025EA 0FB7D0              <1>       movzx  edx, ax
6928                              <1>       ; 08/08/2016
6929 000025ED 0FB605[BE6A0000]    <1>       movzx  eax, byte [CHAR_HEIGHT]
6930 000025F4 F7E2                <1>       mul    edx
6931                              <1>       ; eax = byte count
6932 000025F6 89C1                <1>       mov    ecx, eax
6933                              <1>       ;; 07/08/2016
6934                              <1>       ;shl   dx, 3 ; * 8 ; * [CHAR_HEIGHT]
6935                              <1>       ;mov   ecx, edx
6936 000025F8 88D8                <1>       mov    al, bl ; fill value for blanked lines
6937 000025FA BF00000A00          <1>       mov    edi, 0A0000h
6938 000025FF F3AA                <1>       rep    stosb
6939                              <1>
6940 00002601 66B80500            <1>       mov    ax, 5
6941 00002605 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
6942 00002609 66EF                <1>       out    dx, ax ; 0005h
6943                              <1>
6944 0000260B C3                  <1>       retn
6945                              <1>
6946                              <1> vga_g_up_5:
6947                              <1>       ;;pop  ax
6948                              <1>       ; 12/04/2021
6949                              <1>       ;pop   eax
6950                              <1>
6951                              <1> vga_g_up_6:
6952                              <1>       ; [ESI] = VGA memory model number for current video mode
6953                              <1>         ;
6954                              <1>         ; LINEAR8 equ 5
6955                              <1>         ; PLANAR4 equ 4
6956                              <1>         ; PLANAR1 equ 3
6957                              <1>
6958 0000260C 803E04              <1>       cmp    byte [esi], PLANAR4
6959 0000260F 7424                <1>       je     short vga_g_up_planar
6960 00002611 803E03              <1>       cmp    byte [esi], PLANAR1
6961 00002614 741F                <1>       je     short vga_g_up_planar
6962                              <1> vga_g_up_linear8:
6963                              <1>       ; 07/07/2016 (TEMPORARY)
6964                              <1>       ;
6965                              <1>       ; cl = upper left column ; cul
6966                              <1>       ; ch = upper left row ; rul
6967                              <1>       ; dl = lower rigth column ; clr
6968                              <1>       ; dh = lower right row ; rlr
6969                              <1>
6970                              <1> vga_g_up_l0:
6971                              <1>       ;{for(i=rul;i<=rlr;i++)
6972                              <1>       ; if((i+nblines>rlr)||(nblines==0))
6973 00002616 08C0                <1>       or     al, al
6974 00002618 7414                <1>       jz     short vga_g_up_l2
6975 0000261A 88C4                <1>       mov    ah, al
6976 0000261C 00EC                <1>       add    ah, ch ; i+nblines
6977                              <1>       ;jc    short vga_g_up_l2
6978 0000261E 38F4                <1>       cmp    ah, dh
6979 00002620 770C                <1>       ja     short vga_g_up_l2
6980                              <1>       ; else
6981                              <1>       ;  vgamem_copy_pl4(cul,i+nblines,i,cols,nbcols,cheight);
6982 00002622 E8F2000000          <1>       call   vgamem_copy_l8
6983                              <1> vga_g_up_l1:
6984 00002627 FEC5                <1>       inc    ch
6985 00002629 38F5                <1>       cmp    ch, dh
6986 0000262B 76E9                <1>       jna    short vga_g_up_l0
6987 0000262D C3                  <1>       retn
```

```
                                <1> vga_g_up_l2:
                                <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
0000262E E850010000             <1>       call   vgamem_fill_l8
00002633 EBF2                   <1>       jmp    short vga_g_up_l1
                                <1>
                                <1> vga_g_up_planar:
                                <1>       ; cl = upper left column ; cul
                                <1>       ; ch = upper left row ; rul
                                <1>       ; dl = lower rigth column ; clr
                                <1>       ; dh = lower right row ; rlr
                                <1> vga_g_up_pl0:
                                <1>       ;{for(i=rul;i<=rlr;i++)
                                <1>       ; if((i+nblines>rlr)||(nblines==0))
00002635 20C0                   <1>       and    al, al
00002637 7414                   <1>       jz     short vga_g_up_pl2
00002639 88C4                   <1>       mov    ah, al
0000263B 00EC                   <1>       add    ah, ch ; i+nblines
                                <1>       ;jc    short vga_g_up_pl2
0000263D 38F4                   <1>       cmp    ah, dh
0000263F 770C                   <1>       ja     short vga_g_up_pl2
                                <1>       ; else
                                <1>       ;  vgamem_copy_pl4(cul,i+nblines,i,cols,nbcols,cheight);
00002641 E80E000000             <1>       call   vgamem_copy_pl4
                                <1> vga_g_up_pl1:
00002646 FEC5                   <1>       inc    ch
00002648 38F5                   <1>       cmp    ch, dh
0000264A 76E9                   <1>       jna    short vga_g_up_pl0
0000264C C3                     <1>       retn
                                <1> vga_g_up_pl2:
                                <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
0000264D E870000000             <1>       call   vgamem_fill_pl4
00002652 EBF2                   <1>       jmp    short vga_g_up_pl1
                                <1>
                                <1> vgamem_copy_pl4:
                                <1>       ; 08/08/2016
                                <1>       ; 07/08/2016
                                <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ;
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
                                <1>       ; 'vgabios.c', 'vgamem_copy_pl4'
                                <1>       ;
                                <1>       ; vgamem_copy_pl4(xstart,ysrc,ydest,cols,nbcols,cheight)
                                <1>       ; cl = xstart, ah = ysrc (i+nblines), ch = ydest (i),
                                <1>       ; bh = cols, [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
                                <1>
                                <1>       ; src=ysrc*cheight*nbcols+xstart;
                                <1>       ; dest=ydest*cheight*nbcols+xstart;
                                <1>
00002654 52                     <1>       push   edx
00002655 50                     <1>       push   eax
                                <1>
                                <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0105)
00002656 66B80501               <1>       mov    ax, 0105h
0000265A 66BACE03               <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
0000265E 66EF                   <1>       out    dx, ax
                                <1>
                                <1>       ; 07/08/2016
                                <1>       ;mov     ah, [esp+1]
                                <1>       ;movzx edx, ah ; ysrc
00002660 0FB6542401             <1>       movzx  edx, byte [esp+1]
                                <1>       ; 08/08/2016
00002665 0FB605[BE6A0000]       <1>       movzx  eax, byte [CHAR_HEIGHT]
0000266C 8A25[BC6A0000]         <1>       mov    ah, [CRT_COLS] ; nbcols
00002672 F6E4                   <1>       mul    ah
                                <1>       ;; 07/08/2016
                                <1>       ;movzx eax, byte [CRT_COLS]
                                <1>       ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
00002674 50                     <1>       push   eax ; cheight * nbcols
00002675 F7E2                   <1>       mul    edx ; * ysrc
                                <1>       ; eax = ysrc * cheight * nbcols
                                <1>       ; edx = 0
00002677 88CA                   <1>       mov    dl, cl ; edx = xstart
00002679 01D0                   <1>       add    eax, edx
0000267B 89C6                   <1>       mov    esi, eax ; src
0000267D 88EA                   <1>       mov    dl, ch ; ydest
0000267F 58                     <1>       pop    eax ; cheight * nbcols
00002680 F7E2                   <1>       mul    edx
                                <1>       ; eax = ydest * cheight * nbcols
00002682 88CA                   <1>       mov    dl, cl ; edx = xstart
00002684 01D0                   <1>       add    eax, edx
00002686 89C7                   <1>       mov    edi, eax ; dest
                                <1>       ; esi = src
                                <1>       ; edi = dest
                                <1>       ; for(i=0;i<cheight;i++)
                                <1>       ; {
                                <1>       ;  memcpyb(0xa000,dest+i*nbcols,0xa000,src+i*nbcols,cols);
                                <1>       ; }
00002688 51                     <1>       push   ecx
00002689 B900000A00             <1>       mov    ecx, 0A0000h
0000268E 01CE                   <1>       add    esi, ecx
00002690 01CF                   <1>       add    edi, ecx
                                <1>       ; 08/08/2016
00002692 8A35[BE6A0000]         <1>       mov    dh, [CHAR_HEIGHT]
                                <1>       ;; 07/08/2016
                                <1>       ;mov   dh, 8 ; 07/08/2016
00002698 28D2                   <1>       sub    dl, dl ; i
                                <1> vgamem_copy_pl4_0:
0000269A 56                     <1>       push   esi
0000269B 57                     <1>       push   edi
0000269C 0FB605[BC6A0000]       <1>       movzx  eax, byte [CRT_COLS]
000026A3 F6E2                   <1>       mul    dl
                                <1>       ; eax = i * nbcols
000026A5 01C7                   <1>       add    edi, eax ; dest+i*nbcols
```

```
7093 000026A7 01C6              <1>       add    esi, eax
7094 000026A9 0FB6CF            <1>       movzx  ecx, bh ; cols
7095 000026AC F3A4              <1>       rep    movsb
7096 000026AE 5F                <1>       pop    edi
7097 000026AF 5E                <1>       pop    esi
7098 000026B0 FECE              <1>       dec    dh
7099 000026B2 75E6              <1>       jnz    short vgamem_copy_pl4_0
7100                            <1> vgamem_copy_pl4_1:
7101 000026B4 59                <1>       pop    ecx
7102                            <1>
7103                            <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0005);
7104 000026B5 66B80500          <1>       mov    ax, 0005h
7105 000026B9 66BACE03          <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
7106 000026BD 66EF              <1>       out    dx, ax
7107                            <1>
7108 000026BF 58                <1>       pop    eax
7109 000026C0 5A                <1>       pop    edx
7110                            <1>
7111 000026C1 C3                <1>       retn
7112                            <1>
7113                            <1> vgamem_fill_pl4:
7114                            <1>       ; 08/08/2016
7115                            <1>       ; 07/08/2016
7116                            <1>       ; 04/08/2016
7117                            <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
7118                            <1>       ;
7119                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
7120                            <1>       ; vgabios-0.7a (2011)
7121                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
7122                            <1>       ; 'vgabios.c', 'vgamem_fill_pl4'
7123                            <1>       ;
7124                            <1>       ; vgamem_fill_pl4(xstart,ystart,cols,nbcols,cheight,attr)
7125                            <1>       ; cl = xstart, edi = ch = ystart, bh = cols,
7126                            <1>       ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight, attr = 0
7127                            <1>
7128                            <1>       ; dest=ystart*cheight*nbcols+xstart;
7129 000026C2 52                <1>       push   edx
7130 000026C3 50                <1>       push   eax
7131                            <1>
7132                            <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0205)
7133 000026C4 66B80502          <1>       mov    ax, 0205h
7134 000026C8 66BACE03          <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
7135 000026CC 66EF              <1>       out    dx, ax
7136                            <1>
7137                            <1>              ; 08/08/2016
7138 000026CE 0FB605[BE6A0000]  <1>       movzx   eax, byte [CHAR_HEIGHT]
7139 000026D5 F6E5              <1>       mul    ch
7140                            <1>       ;; 07/08/2016
7141                            <1>       ;movzx eax, ch
7142                            <1>       ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
7143 000026D7 0FB615[BC6A0000]  <1>       movzx  edx, byte [CRT_COLS] ; = [VGA_COLS]
7144 000026DE F7E2              <1>       mul    edx
7145                            <1>       ; edx  = 0
7146 000026E0 88CA              <1>       mov    dl, cl
7147 000026E2 01D0              <1>       add    eax, edx
7148 000026E4 89C7              <1>       mov    edi, eax
7149                            <1>       ; edi = dest
7150                            <1>       ; for(i=0;i<cheight;i++)
7151                            <1>       ; {
7152                            <1>       ;  memsetb(0xa000,dest+i*nbcols,attr,cols);
7153                            <1>       ; }
7154 000026E6 81C700000A00      <1>       add    edi, 0A0000h
7155 000026EC 51                <1>       push   ecx
7156                            <1>       ; 08/08/2016
7157 000026ED 8A35[BE6A0000]    <1>       mov    dh, [CHAR_HEIGHT]
7158                            <1>       ;; 07/08/2016
7159                            <1>       ;mov   dh, 8 ; 07/08/2016
7160 000026F3 28D2              <1>       sub    dl, dl ; i
7161                            <1> vgamem_fill_pl4_0:
7162 000026F5 57                <1>       push   edi
7163 000026F6 0FB605[BC6A0000]  <1>       movzx  eax, byte [CRT_COLS]
7164 000026FD F6E2              <1>       mul    dl
7165                            <1>       ; eax = i * nbcols
7166 000026FF 01C7              <1>       add    edi, eax ; dest+i*nbcols
7167 00002701 88D8              <1>       mov    al, bl ; attr ; 04/08/2016
7168 00002703 0FB6CF            <1>       movzx  ecx, bh ; cols
7169 00002706 F3AA              <1>       rep    stosb
7170 00002708 5F                <1>       pop    edi
7171 00002709 75EA              <1>       jnz    short vgamem_fill_pl4_0
7172                            <1> vgamem_fill_pl4_1:
7173 0000270B 59                <1>       pop    ecx
7174                            <1>
7175                            <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0005);
7176 0000270C 66B80500          <1>       mov    ax, 0005h
7177 00002710 66BACE03          <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
7178 00002714 66EF              <1>       out    dx, ax
7179                            <1>
7180 00002716 58                <1>       pop    eax
7181 00002717 5A                <1>       pop    edx
7182                            <1>
7183 00002718 C3                <1>       retn
7184                            <1>
7185                            <1> vgamem_copy_l8:
7186                            <1>       ; 08/08/2016
7187                            <1>       ; 07/08/2016
7188                            <1>       ; 06/08/2016
7189                            <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
7190                            <1>       ;
7191                            <1>       ; TEMPORARY
7192                            <1>       ;
7193                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
7194                            <1>       ; vgabios-0.7a (2011)
7195                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
7196                            <1>       ; 'vgabios.c', 'vgamem_copy_pl4'
7197                            <1>       ;
```

```
7198                                    <1>     ; vgamem_copy_pl4(xstart,ysrc,ydest,cols,nbcols,cheight)
7199                                    <1>     ; cl = xstart, ah = ysrc (i+nblines), ch = ydest (i),
7200                                    <1>     ; bh = cols, [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
7201                                    <1>
7202                                    <1>     ; src=ysrc*cheight*nbcols+xstart;
7203                                    <1>     ; dest=ydest*cheight*nbcols+xstart;
7204                                    <1>
7205 00002719 52                       <1>     push   edx
7206 0000271A 50                       <1>     push   eax
7207                                    <1>
7208                                    <1>     ; outw(VGAREG_GRDC_ADDRESS, 0x0105)
7209                                    <1>     ;mov   ax, 0105h
7210                                    <1>     ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
7211                                    <1>     ;out   dx, ax
7212                                    <1>
7213                                    <1>     ;mov   ah, [esp+1]
7214                                    <1>
7215 0000271B 0FB6D4                   <1>     movzx  edx, ah ; ysrc
7216                                    <1>     ; 08/08/2016
7217 0000271E 0FB605[BE6A0000]         <1>     movzx  eax, byte [CHAR_HEIGHT]
7218 00002725 8A25[BC6A0000]           <1>     mov    ah, [CRT_COLS] ; nbcols
7219 0000272B F6E4                     <1>     mul    ah
7220                                    <1>     ;; 07/08/2016
7221                                    <1>     ;movzx eax, byte [CRT_COLS]
7222                                    <1>     ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
7223 0000272D 50                       <1>     push   eax ; cheight * nbcols
7224 0000272E F7E2                     <1>     mul    edx ; * ysrc
7225                                    <1>     ; eax = ysrc * cheight * nbcols
7226                                    <1>     ; edx = 0
7227 00002730 88CA                     <1>     mov    dl, cl ; edx = xstart
7228 00002732 01D0                     <1>     add    eax, edx
7229 00002734 89C6                     <1>     mov    esi, eax ; src
7230 00002736 66C1E603                 <1>     shl    si, 3 ; * 8 ; 06/08/2016
7231 0000273A 88EA                     <1>     mov    dl, ch ; ydest
7232 0000273C 58                       <1>     pop    eax ; cheight * nbcols
7233 0000273D F7E2                     <1>     mul    edx
7234                                    <1>     ; eax = ydest * cheight * nbcols
7235 0000273F 88CA                     <1>     mov    dl, cl ; edx = xstart
7236 00002741 01D0                     <1>     add    eax, edx
7237 00002743 89C7                     <1>     mov    edi, eax ; dest
7238 00002745 66C1E703                 <1>     shl    di, 3 ; * 8 ; 06/08/2016
7239                                    <1>     ; esi = src
7240                                    <1>     ; edi = dest
7241                                    <1>     ; for(i=0;i<cheight;i++)
7242                                    <1>     ; {
7243                                    <1>     ;  memcpyb(0xa000,dest+i*nbcols,0xa000,src+i*nbcols,cols);
7244                                    <1>     ; }
7245 00002749 51                       <1>     push   ecx
7246 0000274A B900000A00               <1>     mov    ecx, 0A0000h
7247 0000274F 01CE                     <1>     add    esi, ecx
7248 00002751 01CF                     <1>     add    edi, ecx
7249                                    <1>     ; 08/08/2016
7250 00002753 8A35[BE6A0000]           <1>     mov    dh, [CHAR_HEIGHT]
7251                                    <1>     ;; 07/08/2016
7252                                    <1>     ;mov   dh, 8 ; 07/08/2016
7253 00002759 28D2                     <1>     sub    dl, dl ; i
7254                                    <1> vgamem_copy_l8_0:
7255 0000275B 56                       <1>     push   esi
7256 0000275C 57                       <1>     push   edi
7257 0000275D 0FB605[BC6A0000]         <1>     movzx  eax, byte [CRT_COLS]
7258 00002764 F6E2                     <1>     mul    dl
7259                                    <1>     ; eax = i * nbcols
7260 00002766 66C1E003                 <1>     shl    ax, 3 ; * 8 ; 06/08/2016
7261 0000276A 01C7                     <1>     add    edi, eax ; dest+i*nbcols
7262 0000276C 01C6                     <1>     add    esi, eax
7263 0000276E 0FB6CF                   <1>     movzx  ecx, bh ; cols
7264 00002771 66C1E103                 <1>     shl    cx, 3 ; * 8 ; 06/08/2016
7265 00002775 F3A4                     <1>     rep    movsb
7266 00002777 5F                       <1>     pop    edi
7267 00002778 5E                       <1>     pop    esi
7268 00002779 FEC2                     <1>     inc    dl ; 06/08/2016
7269 0000277B FECE                     <1>     dec    dh
7270 0000277D 75DC                     <1>     jnz    short vgamem_copy_l8_0
7271                                    <1> vgamem_copy_l8_1:
7272 0000277F 59                       <1>     pop    ecx
7273                                    <1>
7274                                    <1>     ;; outw(VGAREG_GRDC_ADDRESS, 0x0005);
7275                                    <1>     ;mov   ax, 0005h
7276                                    <1>     ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
7277                                    <1>     ;out   dx, ax
7278                                    <1>
7279 00002780 58                       <1>     pop    eax
7280 00002781 5A                       <1>     pop    edx
7281                                    <1>
7282 00002782 C3                       <1>     retn
7283                                    <1>
7284                                    <1> vgamem_fill_l8:
7285                                    <1>     ; 08/08/2016
7286                                    <1>     ; 07/08/2016
7287                                    <1>     ; 06/08/2016
7288                                    <1>     ; 04/08/2016
7289                                    <1>     ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
7290                                    <1>     ;
7291                                    <1>     ; TEMPORARY
7292                                    <1>     ;
7293                                    <1>     ; derived from 'Plex86/Bochs VGABios' source code
7294                                    <1>     ; vgabios-0.7a (2011)
7295                                    <1>     ; by the LGPL VGABios developers Team (2001-2008)
7296                                    <1>     ; 'vgabios.c', 'vgamem_fill_pl4'
7297                                    <1>     ;
7298                                    <1>     ; vgamem_fill_pl4(xstart,ystart,cols,nbcols,cheight,attr)
7299                                    <1>     ; cl = xstart, edi = ch = ystart, bh = cols,
7300                                    <1>     ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight, attr = 0
7301                                    <1>
7302                                    <1>     ; dest=ystart*cheight*nbcols+xstart;
```

```
7303 00002783 52                  <1>       push   edx
7304 00002784 50                  <1>       push   eax
7305                              <1>
7306                              <1>       ;; outw(VGAREG_GRDC_ADDRESS, 0x0205)
7307                              <1>       ;mov   ax, 0205h
7308                              <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
7309                              <1>       ;out   dx, ax
7310                              <1>
7311                              <1>         ; 08/08/2016
7312 00002785 0FB605[BE6A0000]    <1>       movzx   eax, byte [CHAR_HEIGHT]
7313 0000278C F6E5                <1>       mul    ch
7314                              <1>       ;; 07/08/2016
7315                              <1>       ;movzx eax, ch
7316                              <1>       ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
7317 0000278E 0FB615[BC6A0000]    <1>       movzx  edx, byte [CRT_COLS] ; = [VGA_COLS]
7318 00002795 F7E2                <1>       mul    edx
7319                              <1>       ; edx  = 0
7320 00002797 88CA                <1>       mov    dl, cl
7321 00002799 01D0                <1>       add    eax, edx
7322 0000279B 89C7                <1>       mov    edi, eax
7323 0000279D 66C1E703            <1>       shl    di, 3 ; * 8 ; 06/08/2016
7324                              <1>       ; edi = dest
7325                              <1>       ; for(i=0;i<cheight;i++)
7326                              <1>       ; {
7327                              <1>       ;  memsetb(0xa000,dest+i*nbcols,attr,cols);
7328                              <1>       ; }
7329 000027A1 81C700000A00        <1>       add    edi, 0A0000h
7330 000027A7 51                  <1>       push   ecx
7331                              <1>       ; 08/08/2016
7332 000027A8 8A35[BE6A0000]      <1>       mov    dh, [CHAR_HEIGHT]
7333                              <1>       ;; 07/08/2016
7334                              <1>       ;mov   dh, 8 ; 07/08/2016
7335 000027AE 28D2                <1>       sub    dl, dl ; i
7336                              <1> vgamem_fill_18_0:
7337 000027B0 57                  <1>       push   edi
7338 000027B1 0FB605[BC6A0000]    <1>       movzx  eax, byte [CRT_COLS]
7339 000027B8 F6E2                <1>       mul    dl
7340                              <1>       ; eax = i * nbcols
7341 000027BA 66C1E003            <1>       shl    ax, 3 ; * 8 ; 06/08/2016
7342 000027BE 01C7                <1>       add    edi, eax ; dest+i*nbcols
7343 000027C0 88D8                <1>       mov    al, bl ; attr ; 04/08/2016
7344 000027C2 0FB6CF              <1>       movzx  ecx, bh ; cols
7345 000027C5 66C1E103            <1>       shl    cx, 3 ; * 8 ; 06/08/2016
7346 000027C9 F3AA                <1>       rep    stosb
7347 000027CB 5F                  <1>       pop    edi
7348 000027CC FEC2                <1>       inc    dl ; 06/08/2016
7349 000027CE FECE                <1>       dec    dh
7350 000027D0 75DE                <1>       jnz    short vgamem_fill_18_0
7351                              <1> vgamem_fill_18_1:
7352 000027D2 59                  <1>       pop    ecx
7353                              <1>
7354                              <1>       ;; outw(VGAREG_GRDC_ADDRESS, 0x0005);
7355                              <1>       ;mov   ax, 0005h
7356                              <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
7357                              <1>       ;out   dx, ax
7358                              <1>
7359 000027D3 58                  <1>       pop    eax
7360 000027D4 5A                  <1>       pop    edx
7361                              <1>
7362 000027D5 C3                  <1>       retn
7363                              <1>
7364                              <1> vga_graphics_down:
7365                              <1>       ; 12/04/2021
7366                              <1>       ; 08/08/2016
7367                              <1>       ; 07/08/2016
7368                              <1>       ; 31/07/2016
7369                              <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
7370                              <1>       ;
7371                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
7372                              <1>       ; vgabios-0.7a (2011)
7373                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
7374                              <1>       ; 'vgabios.c', 'biosfn_scroll'
7375                              <1>       ;
7376                              <1>
7377                              <1>       ; cl = upper left column
7378                              <1>       ; ch = upper left row
7379                              <1>       ; dl = lower rigth column
7380                              <1>       ; dh = lower right row
7381                              <1>       ;
7382                              <1>       ; al = line count (AL=0 means blank entire fields)
7383                              <1>       ; bl = fill value for blanked lines
7384                              <1>       ; bh = unused
7385                              <1>       ;
7386                              <1>       ; ah = [CRT_MODE], current video mode
7387                              <1>
7388 000027D6 FC                  <1>         cld     ; !!! Clear direction flag !!!
7389                              <1>
7390 000027D7 88C7                <1>       mov    bh, al ; 31/07/2016
7391                              <1>
7392 000027D9 BE[D66A0000]        <1>       mov    esi, vga_modes
7393 000027DE 89F7                <1>       mov    edi, esi
7394 000027E0 83C710              <1>       add    edi, vga_mode_count
7395                              <1> vga_g_down_0:
7396 000027E3 AC                  <1>       lodsb
7397 000027E4 38E0                <1>       cmp    al, ah ; [CRT_MODE]
7398 000027E6 7405                <1>       je     short vga_g_down_1
7399 000027E8 39FE                <1>       cmp    esi, edi
7400 000027EA 72F7                <1>       jb     short vga_g_down_0
7401                              <1>       ; xor  bh, bh ; 31/07/2016
7402 000027EC C3                  <1>       retn   ; nothing to do
7403                              <1> vga_g_down_1:
7404 000027ED 88F8                <1>       mov    al, bh ; 31/07/2016
7405 000027EF 83C64F              <1>       add    esi, vga_memmodel - (vga_modes + 1)
7406                              <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
7407                              <1>
```

```
7408                                    <1>       ; if(rlr>=nbrows)rlr=nbrows-1;
7409                                    <1>       ; if(clr>=nbcols)clr=nbcols-1;
7410                                    <1>       ; if(nblines>nbrows)nblines=0;
7411                                    <1>       ; cols=clr-cul+1;
7412                                    <1>
7413 000027F2 3A35[C26A0000]            <1>       cmp    dh, [VGA_ROWS]
7414 000027F8 7208                     <1>       jb     short vga_g_down_2
7415 000027FA 8A35[C26A0000]           <1>       mov    dh, [VGA_ROWS]
7416 00002800 FECE                     <1>       dec    dh
7417                                    <1> vga_g_down_2:
7418 00002802 3A15[BC6A0000]           <1>       cmp    dl, [CRT_COLS]  ; = [VGA_COLS]
7419 00002808 7208                     <1>       jb     short vga_g_down_3
7420 0000280A 8A15[BC6A0000]           <1>       mov    dl, [CRT_COLS]
7421 00002810 FECA                     <1>       dec    dl
7422                                    <1> vga_g_down_3:
7423 00002812 3A05[C26A0000]           <1>       cmp    al, [VGA_ROWS]
7424 00002818 7602                     <1>       jna    short vga_g_down_4
7425 0000281A 28C0                     <1>       sub    al, al ; 0
7426                                    <1> vga_g_down_4:
7427 0000281C 88F7                     <1>       mov    bh, dh ; clr
7428 0000281E 28CF                     <1>       sub    bh, cl ; cul
7429 00002820 FEC7                     <1>       inc    bh ; cols = clr-cul+1
7430                                    <1>
7431 00002822 20C0                     <1>       and    al, al ; nblines = 0
7432 00002824 7557                     <1>       jnz    short vga_g_down_6
7433 00002826 20ED                     <1>       and    ch, ch ; rul = 0
7434 00002828 7553                     <1>       jnz    short vga_g_down_6
7435 0000282A 20C9                     <1>       and    cl, cl ; cul = 0
7436 0000282C 754F                     <1>       jnz    short vga_g_down_6
7437                                    <1>
7438 0000282E 50                       <1>       push   eax ; push ax ; 12/04/2021
7439 0000282F A0[C26A0000]             <1>       mov    al, [VGA_ROWS]
7440 00002834 FEC8                     <1>       dec    al
7441 00002836 38C6                     <1>       cmp    dh, al ; rlr = nbrows-1
7442 00002838 7543                     <1>       jne    short vga_g_down_5
7443 0000283A A0[BC6A0000]             <1>         mov     al, [CRT_COLS]  ; = VGA_COLS
7444 0000283F FEC8                     <1>       dec    al
7445 00002841 38C2                     <1>       cmp    dl, al ; clr = nbcols-1
7446                                    <1>       ;jne   short vga_g_down_5
7447                                    <1>       ; 12/04/2021
7448 00002843 58                       <1>       pop    eax ; pop ax
7449 00002844 7537                     <1>       jne    short vga_g_down_5
7450                                    <1>
7451 00002846 66B80502                 <1>       mov    ax, 0205h
7452 0000284A 66BACE03                 <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
7453 0000284E 66EF                     <1>       out    dx, ax
7454 00002850 A0[C26A0000]             <1>       mov    al, [VGA_ROWS]
7455 00002855 8A25[BC6A0000]           <1>       mov    ah, [CRT_COLS] ; = [VGA_COLS]
7456 0000285B F6E4                     <1>       mul    ah
7457 0000285D 0FB7D0                   <1>       movzx  edx, ax
7458                                    <1>       ; 08/08/2016
7459 00002860 0FB605[BE6A0000]         <1>       movzx  eax, byte [CHAR_HEIGHT]
7460 00002867 F7E2                     <1>       mul    edx
7461                                    <1>       ; eax = byte count
7462 00002869 89C1                     <1>       mov    ecx, eax
7463                                    <1>       ;; 07/08/2016
7464                                    <1>       ;shl   dx, 3 ; * 8 ; * [CHAR_HEIGHT]
7465                                    <1>       ;mov   ecx, edx
7466 0000286B 88D8                     <1>       mov    al, bl ; fill value for blanked lines
7467 0000286D BF00000A00               <1>       mov    edi, 0A0000h
7468 00002872 F3AA                     <1>       rep    stosb
7469                                    <1>
7470 00002874 B005                     <1>       mov    al, 5
7471 00002876 66BACE03                 <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
7472 0000287A 66EF                     <1>       out    dx, ax ; 0005h
7473                                    <1>
7474 0000287C C3                       <1>       retn
7475                                    <1>
7476                                    <1> vga_g_down_5:
7477                                    <1>       ; 12/04/2021
7478                                    <1>       ;pop   eax  ; pop ax
7479                                    <1>
7480                                    <1> vga_g_down_6:
7481                                    <1>       ; [ESI] = VGA memory model number for current video mode
7482                                    <1>         ;
7483                                    <1>         ; LINEAR8 equ 5
7484                                    <1>         ; PLANAR4 equ 4
7485                                    <1>         ; PLANAR1 equ 3
7486                                    <1>
7487 0000287D 803E04                   <1>       cmp    byte [esi], PLANAR4
7488 00002880 742C                     <1>       je     short vga_g_down_planar
7489 00002882 803E03                   <1>       cmp    byte [esi], PLANAR1
7490 00002885 7427                     <1>       je     short vga_g_down_planar
7491                                    <1> vga_g_down_linear8:
7492                                    <1>       ; 07/07/2016 (TEMPORARY)
7493                                    <1>       ;
7494                                    <1>       ; cl = upper left column ; cul
7495                                    <1>       ; ch = upper left row ; rul
7496                                    <1>       ; dl = lower rigth column ; clr
7497                                    <1>       ; dh = lower right row ; rlr
7498                                    <1>
7499                                    <1> vga_g_down_l0:
7500                                    <1>       ;{for(i=rlr;i>=rul;i--)
7501                                    <1>         ; if((i<rul+nblines)||(nblines==0))
7502 00002887 08C0                     <1>       or     al, al
7503 00002889 741C                     <1>       jz     short vga_g_down_l2
7504 0000288B 88C4                     <1>       mov    ah, al
7505 0000288D 00EC                     <1>       add    ah, ch
7506                                    <1>       ;jc    short vga_g_down_l2
7507 0000288F 86EE                     <1>       xchg   ch, dh
7508 00002891 38E5                     <1>       cmp    ch, ah
7509 00002893 7212                     <1>       jb     short vga_g_down_l2
7510 00002895 88EC                     <1>       mov    ah, ch
7511 00002897 28C4                     <1>       sub    ah, al ; ah = i - nblines
7512                                    <1>       ; else
```

```
7513                                    <1>       ; vgamem_copy_pl4(cul,i,i-nblines,cols,nbcols,cheight);
7514 00002899 E87BFEFFFF                <1>       call   vgamem_copy_l8
7515                                    <1> vga_g_down_l1:
7516 0000289E 86F5                      <1>       xchg   dh, ch
7517 000028A0 FECE                      <1>       dec    dh
7518 000028A2 38EE                      <1>       cmp    dh, ch
7519 000028A4 73E1                      <1>       jnb    short vga_g_down_l0
7520 000028A6 C3                        <1>       retn
7521                                    <1>
7522                                    <1> vga_g_down_l2:
7523                                    <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
7524 000028A7 E8D7FEFFFF                <1>       call   vgamem_fill_l8
7525 000028AC EBF0                      <1>       jmp    short vga_g_down_l1
7526                                    <1>
7527                                    <1> vga_g_down_planar:
7528                                    <1>       ; cl = upper left column ; cul
7529                                    <1>       ; ch = upper left row ; rul
7530                                    <1>       ; dl = lower rigth column ; clr
7531                                    <1>       ; dh = lower right row ; rlr
7532                                    <1> vga_g_down_pl0:
7533                                    <1>       ;{for(i=rlr;i>=rul;i--)
7534                                    <1>         ; if((i<rul+nblines)||(nblines==0))
7535 000028AE 08C0                      <1>       or     al, al
7536 000028B0 741C                      <1>       jz     short vga_g_down_pl2
7537 000028B2 88C4                      <1>       mov    ah, al
7538 000028B4 00EC                      <1>       add    ah, ch
7539                                    <1>       ;jc    short vga_g_down_pl2
7540 000028B6 86EE                      <1>       xchg   ch, dh
7541 000028B8 38E5                      <1>       cmp    ch, ah
7542 000028BA 7212                      <1>       jb     short vga_g_down_pl2
7543 000028BC 88EC                      <1>       mov    ah, ch
7544 000028BE 28C4                      <1>       sub    ah, al ; ah = i - nblines
7545                                    <1>       ; else
7546                                    <1>       ; vgamem_copy_pl4(cul,i,i-nblines,cols,nbcols,cheight);
7547 000028C0 E88FFDFFFF                <1>       call   vgamem_copy_pl4
7548                                    <1> vga_g_down_pl1:
7549 000028C5 86F5                      <1>       xchg   dh, ch
7550 000028C7 FECE                      <1>       dec    dh
7551 000028C9 38EE                      <1>       cmp    dh, ch
7552 000028CB 73E1                      <1>       jnb    short vga_g_down_pl0
7553 000028CD C3                        <1>       retn
7554                                    <1>
7555                                    <1> vga_g_down_pl2:
7556                                    <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
7557 000028CE E8EFFDFFFF                <1>       call   vgamem_fill_pl4
7558 000028D3 EBF0                      <1>       jmp    short vga_g_down_pl1
7559                                    <1>
7560                                    <1> ; 07/07/2016
7561                                    <1> ; 27/06/2016 - TRDOS 386 (TRDOS v2.0)
7562                                    <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7563                                    <1> ;--------------------------------------------------------
7564                                    <1> ; SCROLL DOWN
7565                                    <1> ;  THIS ROUTINE SCROLLS DOWN THE INFORMATION ON THE CRT
7566                                    <1> ; ENTRY --
7567                                    <1> ;  CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL
7568                                    <1> ;  DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL
7569                                    <1> ;   BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS
7570                                    <1> ;  BH = FILL VALUE FOR BLANKED LINES
7571                                    <1> ;  AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE FIELD)
7572                                    <1> ;  DS = DATA SEGMENT
7573                                    <1> ;  ES = REGEN SEGMENT
7574                                    <1> ; EXIT --
7575                                    <1> ;  NOTHING, THE SCREEN IS SCROLLED
7576                                    <1> ;--------------------------------------------------------
7577                                    <1>
7578                                    <1>       ; cl = upper left column
7579                                    <1>       ; ch = upper left row
7580                                    <1>       ; dl = lower rigth column
7581                                    <1>       ; dh = lower right row
7582                                    <1>       ;
7583                                    <1>       ; al = line count (AL=0 means blank entire fields)
7584                                    <1>       ; bl = fill value for blanked lines
7585                                    <1>       ; bh = unused
7586                                    <1>
7587                                    <1> GRAPHICS_DOWN:
7588                                    <1>       ; 07/07/2016
7589                                    <1>       ;AH = Current video mode, [CRT_MODE]
7590                                    <1>       ;STD                    ; SET DIRECTION
7591 000028D5 80FC07                   <1>       cmp    ah, 7
7592 000028D8 0F87F8FEFFFF             <1>         ja     vga_graphics_down
7593                                    <1>       ;je    _n0
7594                                    <1>
7595 000028DE 88C7                     <1>       MOV    bh, al             ; save line count in BH
7596 000028E0 6689D0                   <1>       MOV    AX, DX             ; GET LOWER RIGHT POSITION INTO AX REG
7597                                    <1>
7598                                    <1> ;-----      USE CHARACTER SUBROUTINE FOR POSITIONING
7599                                    <1> ;-----      ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE
7600                                    <1>
7601 000028E3 E8F0010000               <1>       CALL   GRAPH_POSN
7602 000028E8 0FB7F8                   <1>       MOVzx  eDI, AX                  ; SAVE RESULT AS DESTINATION ADDRESS
7603                                    <1>
7604                                    <1> ;-----      DETERMINE SIZE OF WINDOW
7605                                    <1>
7606 000028EB 6629CA                   <1>       SUB    DX, CX
7607 000028EE 6681C20101               <1>         ADD    DX, 101h               ; ADJUST VALUES
7608 000028F3 C0E602                   <1>       SAL    DH, 2              ; MULTIPLY ROWS BY 4 AT 8 VERT DOTS/CHAR
7609                                    <1>                               ; AND EVEN/ODD ROWS
7610                                    <1>
7611                                    <1> ;-----      DETERMINE CRT MODE
7612                                    <1>
7613 000028F6 803D[BA6A0000]06         <1>       CMP    byte [CRT_MODE], 6 ; TEST FOR MEDIUM RES
7614 000028FD 7307                     <1>         JNC    short _R12             ; FIND_SOURCE_DOWN
7615                                    <1>
7616                                    <1> ;-----      MEDIUM RES DOWN
7617 000028FF D0E2                     <1>       SAL    DL, 1              ; # COLUMNS * 2, SINCE 2 BYTES/CHAR
```

```
7618 00002901 66D1E7              <1>       SAL    DI, 1               ; OFFSET *2 SINCE 2 BYTES/CHAR
7619 00002904 6647                <1>       INC    DI                  ; POINT TO LAST BYTE
7620                              <1>
7621                              <1> ;-----      DETERMINE THE SOURCE ADDRESS IN THE BUFFER
7622                              <1>
7623                              <1> _R12:                                  ; FIND_SOURCE_DOWN
7624 00002906 81C700800B00        <1>       add    edi, 0B8000h
7625 0000290C 6681C7F000          <1>       ADD    DI, 240                 ; POINT TO LAST ROW OF PIXELS
7626 00002911 C0E702              <1>       sal    bh, 2               ; multiply number of lines by 4
7627 00002914 74(06)              <1>       JZ     short 6                 ; IF ZERO, THEN BLANK ENTIRE FIELD
7628 00002916 B050                <1>       MOV    AL, 80              ; 80 BYTES/ROW
7629 00002918 F6E7                <1>       mul    bh                  ; determine offset to source
7630 0000291A 89FE                <1>       MOV    eSI, eDI            ; SET UP SOURCE
7631 0000291C 6629C6              <1>       SUB    SI, AX              ; SUBTRACT THE OFFSET
7632 0000291F 88F4                <1>       MOV    AH, DH              ; NUMBER OF ROWS IN FIELD
7633 00002921 28FC                <1>       sub    ah, bh              ; determine number to move
7634                              <1>
7635                              <1> ;-----      LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS
7636                              <1>
7637                              <1> _R13:                                  ; ROW_LOOP_DOWN
7638 00002923 E824000000          <1>         CALL    _R17                  ; MOVE ONE ROW
7639 00002928 6681EE5020          <1>       SUB    SI, 2000h+80        ; MOVE TO NEXT ROW
7640 0000292D 6681EF5020          <1>       SUB    DI, 2000h+80
7641 00002932 FECC                <1>       DEC    AH                  ; NUMBER OF ROWS TO MOVE
7642 00002934 75ED                <1>         JNZ     short _R13             ; CONTINUE TILL ALL MOVED
7643                              <1>
7644                              <1> ;-----      FILL IN THE VACATED LINE(S)
7645                              <1> _R14:                                  ; CLEAR_ENTRY_DOWN
7646 00002936 88D8                <1>       mov    al, bl              ; attribute to fill with
7647                              <1> _R15_:                                 ; CLEAR_LOOP_DOWN
7648 00002938 E82B000000          <1>       CALL   _R18                ; CLEAR A ROW
7649 0000293D 6681EF5020          <1>       SUB    DI, 2000h+80        ; POINT TO NEXT LINE
7650 00002942 FECF                <1>       dec    bh                  ; number of lines to fill
7651 00002944 75F2                <1>         JNZ     short _R15_            ; CLEAR_LOOP_DOWN
7652                              <1>       ; 18/11/2020
7653 00002946 FC                  <1>       CLD                        ; RESET THE DIRECTION FLAG
7654                              <1>
7655 00002947 C3                  <1>       retn                       ; EVERYTHING DONE
7656                              <1>
7657                              <1> _R16:                                  ; BLANK_FIELD_DOWN
7658 00002948 88F7                <1>       mov    bh, dh              ; set blank count to everything in field
7659 0000294A EBEA                <1>         JMP     short _R14             ; CLEAR THE FIELD
7660                              <1>
7661                              <1> ; 27/06/2016 - TRDOS 386 (TRDOS v2.0)
7662                              <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7663                              <1>
7664                              <1> ;-----      ROUTINE TO MOVE ONE ROW OF INFORMATION
7665                              <1>
7666                              <1> _R17:
7667 0000294C 0FB6CA              <1>       MOVzx  ecx, DL                   ; NUMBER OF BYTES IN THE ROW
7668 0000294F 56                  <1>       PUSH   eSI
7669 00002950 57                  <1>       PUSH   eDI                 ; SAVE POINTERS
7670 00002951 F3A4                <1>       REP    MOVSB               ; MOVE THE EVEN FIELD
7671 00002953 5F                  <1>       POP    eDI
7672 00002954 5E                  <1>       POP    eSI
7673 00002955 6681C60020          <1>       ADD    SI, 2000h
7674 0000295A 6681C70020          <1>       ADD    DI, 2000h           ; POINT TO THE ODD FIELD
7675 0000295F 56                  <1>       PUSH   eSI
7676 00002960 57                  <1>       PUSH   eDI                 ; SAVE THE POINTERS
7677 00002961 88D1                <1>       MOV    CL, DL              ; COUNT BACK
7678 00002963 F3A4                <1>       REP    MOVSB               ; MOVE THE ODD FIELD
7679 00002965 5F                  <1>       POP    eDI
7680 00002966 5E                  <1>       POP    eSI                 ; POINTERS BACK
7681 00002967 C3                  <1>       RETn                       ; RETURN TO CALLER
7682                              <1>
7683                              <1> ;-----      CLEAR A SINGLE ROW
7684                              <1>
7685                              <1> _R18:
7686 00002968 0FB6CA              <1>       MOVzx  ecx, DL                   ; NUMBER OF BYTES IN FIELD
7687 0000296B 57                  <1>       PUSH   eDI                 ; SAVE POINTER
7688 0000296C F3AA                <1>       REP    STOSB               ; STORE THE NEW VALUE
7689 0000296E 5F                  <1>       POP    eDI                 ; POINTER BACK
7690 0000296F 6681C70020          <1>       ADD    DI, 2000h           ; POINT TO ODD FIELD
7691 00002974 57                  <1>       PUSH   eDI
7692 00002975 88D1                <1>       MOV    CL, DL
7693 00002977 F3AA                <1>       REP    STOSB               ; FILL THE ODD FIELD
7694 00002979 5F                  <1>       POP    eDI
7695 0000297A C3                  <1>       RETn                       ; RETURN TO CALLER
7696                              <1>
7697                              <1> ; 04/07/2016
7698                              <1> ; 01/07/2016
7699                              <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
7700                              <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7701                              <1> ;-------------------------------------------------
7702                              <1> ; GRAPHICS WRITE
7703                              <1> ;  THIS ROUTINE WRITES THE ASCII CHARACTER TO THE CURRENT
7704                              <1> ;  POSITION ON THE SCREEN.
7705                              <1> ; ENTRY --
7706                              <1> ;  AL = CHARACTER TO WRITE
7707                              <1> ;  BL = COLOR ATTRIBUTE TO BE USED FOR FOREGROUND COLOR
7708                              <1> ;     IF BIT 7 IS SET, THE CHAR IS XOR'D INTO THE REGEN BUFFER
7709                              <1> ;     (0 IS USED FOR THE BACKGROUND COLOR)
7710                              <1> ;  CX = NUMBER OF CHARS TO WRITE
7711                              <1> ;  DS = DATA SEGMENT
7712                              <1> ;  ES = REGEN SEGMENT
7713                              <1> ; EXIT --
7714                              <1> ;  NOTHING IS RETURNED
7715                              <1> ;
7716                              <1> ; GRAPHICS READ
7717                              <1> ;  THIS ROUTINE READS THE ASCII CHARACTER AT THE CURRENT CURSOR
7718                              <1> ;  POSITION ON THE SCREEN BY MATCHING THE DOTS ON THE SCREEN TO THE
7719                              <1> ;  CHARACTER GENERATOR CODE POINTS
7720                              <1> ; ENTRY --
7721                              <1> ;  NONE (0 IS ASSUMED AS THE BACKGROUND COLOR)
7722                              <1> ; EXIT --
```

```
7723                            <1> ;  AL = CHARACTER READ AT THAT POSITION (0 RETURNED IF NONE FOUND)
7724                            <1> ;
7725                            <1> ; FOR BOTH ROUTINES, THE IMAGES USED TO FORM CHARS ARE CONTAINED IN ROM
7726                            <1> ;  FOR THE 1ST 128 CHARS.  TO ACCESS CHARS IN THE SECOND HALF, THE USER
7727                            <1> ;  MUST INITIALIZE THE VECTOR AT INTERRUPT 1FH (LOCATION 0007CH) TO
7728                            <1> ;  POINT TO THE USER SUPPLIED TABLE OF GRAPHIC IMAGES (8X8 BOXES).
7729                            <1> ;  FAILURE TO DO SO WILL CAUSE IN STRANGE RESULTS
7730                            <1> ;--------------------------------------------------
7731                            <1>
7732                            <1> GRAPHICS_WRITE:
7733 0000297B 25FF000000       <1>       and   eax, 0FFh           ; ZERO TO HIGH OF CODE POINT
7734 00002980 50               <1>       PUSH  eAX                 ; SAVE CODE POINT VALUE
7735                            <1>
7736                            <1> ;-----      DETERMINE POSITION IN REGEN BUFFER TO PUT CODE POINTS
7737                            <1>
7738 00002981 E84B010000       <1>       CALL  S26                 ; FIND LOCATION IN REGEN BUFFER
7739 00002986 89C7             <1>       MOV   eDI, eAX            ; REGEN POINTER IN DI
7740                            <1>
7741                            <1> ;-----      DETERMINE REGION TO GET CODE POINTS FROM
7742                            <1>
7743 00002988 58               <1>       POP   eAX                 ; RECOVER CODE POINT
7744                            <1>
7745 00002989 BE[3C570100]     <1>       MOV   eSI, CRT_CHAR_GEN   ; OFFSET OF IMAGES
7746                            <1>
7747                            <1> ;-----      DETERMINE GRAPHICS MODE IN OPERATION
7748                            <1>                                 ; DETERMINE_MODE
7749 0000298E 66C1E003         <1>       SAL   AX, 3               ; MULTIPLY CODE POINT VALUE BY 8
7750 00002992 01C6             <1>       ADD   eSI, eAX            ; SI HAS OFFSET OF DESIRED CODES
7751                            <1>
7752 00002994 803D[BA6A0000]06 <1>       CMP   byte [CRT_MODE], 6
7753 0000299B 7231             <1>       JC    short S6            ; TEST FOR MEDIUM RESOLUTION MODE
7754                            <1>
7755                            <1> ;-----      HIGH RESOLUTION MODE
7756                            <1>
7757 0000299D 81C700800B00     <1>       add   edi, 0B8000h
7758                            <1> S1:                             ; HIGH_CHAR
7759 000029A3 57               <1>       PUSH  eDI                 ; SAVE REGEN POINTER
7760 000029A4 56               <1>       PUSH  eSI                 ; SAVE CODE POINTER
7761 000029A5 B604             <1>       MOV   DH, 4               ; NUMBER OF TIMES THROUGH LOOP
7762                            <1> S2:
7763 000029A7 AC               <1>       LODSB                     ; GET BYTE FROM CODE POINTS
7764 000029A8 F6C380           <1>       TEST  BL, 80H                   ; SHOULD WE USE THE FUNCTION
7765 000029AB 7515             <1>       JNZ   short S5            ; TO PUT CHAR IN
7766 000029AD AA               <1>       STOSB                     ; STORE IN REGEN BUFFER
7767 000029AE AC               <1>       LODSB
7768                            <1> S4:
7769 000029AF 8887FF1F0000     <1>       MOV   [eDI+2000H-1], AL ; STORE IN SECOND HALF
7770 000029B5 83C74F           <1>       ADD   eDI, 79                   ; MOVE TO NEXT ROW IN REGEN
7771 000029B8 FECE             <1>       DEC   DH                  ; DONE WITH LOOP
7772 000029BA 75EB             <1>       JNZ   short S2
7773 000029BC 5E               <1>       POP   eSI
7774 000029BD 5F               <1>       POP   eDI                 ; RECOVER REGEN POINTER
7775 000029BE 47               <1>       INC   eDI                 ; POINT TO NEXT CHAR POSITION
7776 000029BF E2E2             <1>       LOOP  S1                  ; MORE CHARS TO WRITE
7777 000029C1 C3               <1>       retn
7778                            <1>
7779                            <1> S5:
7780 000029C2 3207             <1>       XOR   AL, [eDI]           ; EXCLUSIVE OR WITH CURRENT
7781 000029C4 AA               <1>       STOSB                     ; STORE THE CODE POINT
7782 000029C5 AC               <1>       LODSB                     ; AGAIN FOR ODD FIELD
7783 000029C6 3287FF1F0000     <1>       XOR   AL, [eDI+2000H-1]
7784 000029CC EBE1             <1>       JMP   short S4            ; BACK TO MAINSTREAM
7785                            <1>
7786                            <1> ;-----      MEDIUM RESOLUTION WRITE
7787                            <1> S6:                             ; MED_RES_WRITE
7788 000029CE 88DA             <1>       MOV   DL, BL              ; SAVE HIGH COLOR BIT
7789 000029D0 66D1E7           <1>       SAL   DI, 1               ; OFFSET*2 SINCE 2 BYTES/CHAR
7790                            <1>                                 ; EXPAND BL TO FULL WORD OF COLOR
7791 000029D3 80E303           <1>       AND   BL, 3               ; ISOLATE THE COLOR BITS ( LOW 2 BITS )
7792 000029D6 B055             <1>       MOV   AL, 055H            ; GET BIT CONVERSION MULTIPLIER
7793 000029D8 F6E3             <1>       MUL   BL                  ; EXPAND 2 COLOR BITS TO 4 REPLICATIONS
7794 000029DA 88C3             <1>       MOV   BL, AL              ; PLACE BACK IN WORK REGISTER
7795 000029DC 88C7             <1>       MOV   BH, AL              ; EXPAND TO 8 REPLICATIONS OF COLOR BITS
7796 000029DE 81C700800B00     <1>       add   edi, 0B8000h
7797                            <1> S7:                                   ; MED_CHAR
7798 000029E4 57               <1>       PUSH  eDI                 ; SAVE REGEN POINTER
7799 000029E5 56               <1>       PUSH  eSI                 ; SAVE THE CODE POINTER
7800 000029E6 B604             <1>       MOV   DH, 4               ; NUMBER OF LOOPS
7801                            <1> S8:
7802 000029E8 AC               <1>       LODSB                     ; GET CODE POINT
7803 000029E9 E8B3000000       <1>       CALL  S21                 ; DOUBLE UP ALL THE BITS
7804 000029EE 6621D8           <1>       AND   AX, BX              ; CONVERT TO FOREGROUND COLOR ( 0 BACK )
7805 000029F1 86E0             <1>       XCHG  AH, AL              ; SWAP HIGH/LOW BYTES FOR WORD MOVE
7806 000029F3 F6C280           <1>       TEST  DL, 80H                   ; IS THIS XOR FUNCTION
7807 000029F6 7403             <1>       JZ    short S9            ; NO, STORE IT IN AS IS
7808 000029F8 663307           <1>       XOR   AX, [eDI]           ; DO FUNCTION WITH LOW/HIGH
7809                            <1> S9:
7810 000029FB 668907           <1>       MOV   [eDI], AX           ; STORE FIRST BYTE HIGH, SECOND LOW
7811 000029FE AC               <1>       LODSB                     ; GET CODE POINT
7812 000029FF E89D000000       <1>       CALL  S21
7813 00002A04 6621D8           <1>       AND   AX, BX              ; CONVERT TO COLOR
7814 00002A07 86E0             <1>       XCHG  AH, AL              ; SWAP HIGH/LOW BYTES FOR WORD MOVE
7815 00002A09 F6C280           <1>       TEST  DL, 80H                   ; AGAIN, IS THIS XOR FUNCTION
7816 00002A0C 7407             <1>       JZ    short _S10          ; NO, JUST STORE THE VALUES
7817 00002A0E 66338700200000   <1>       XOR   AX, [eDI+2000H]           ; FUNCTION WITH FIRST HALF LOW
7818                            <1> _S10:
7819 00002A15 66898700200000   <1>       MOV   [eDI+2000H], AX           ; STORE SECOND PORTION HIGH
7820 00002A1C 6683C750         <1>       ADD   DI, 80              ; POINT TO NEXT LOCATION
7821 00002A20 FECE             <1>       DEC   DH
7822 00002A22 75C4             <1>       JNZ   short S8            ; KEEP GOING
7823 00002A24 5E               <1>       POP   eSI                 ; RECOVER CODE POINTER
7824 00002A25 5F               <1>       POP   eDI                 ; RECOVER REGEN POINTER
7825 00002A26 47               <1>       INC   eDI                 ; POINT TO NEXT CHAR POSITION
7826 00002A27 47               <1>       INC   eDI
7827 00002A28 E2BA             <1>       LOOP  S7                  ; MORE TO WRITE
```

```
7828 00002A2A C3                <1>       retn
7829                            <1>
7830                            <1> ; 04/07/2016
7831                            <1> ; 01/07/2016
7832                            <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
7833                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7834                            <1> ;----------------------------------------
7835                            <1> ; GRAPHICS READ
7836                            <1> ;----------------------------------------
7837                            <1> GRAPHICS_READ:
7838 00002A2B E8A1000000        <1>       CALL   S26                ; CONVERTED TO OFFSET IN REGEN
7839 00002A30 89C6              <1>       MOV    eSI, eAX           ; SAVE IN SI
7840 00002A32 81C600800B00      <1>       add    esi, 0B8000h       ; 01/07/2016
7841 00002A38 83EC08            <1>       SUB    eSP, 8             ; ALLOCATE SPACE FOR THE READ CODE POINT
7842 00002A3B 89E5              <1>       MOV    eBP, eSP           ; POINTER TO SAVE AREA
7843                            <1>
7844                            <1> ;-----       DETERMINE GRAPHICS MODES
7845 00002A3D B604              <1>       mov    dh, 4              ; number of passes ; 01/07/2016
7846 00002A3F 803D[BA6A0000]06  <1>       CMP    byte [CRT_MODE], 6
7847 00002A46 7219              <1>       JC     short S12          ; MEDIUM RESOLUTION
7848                            <1>
7849                            <1> ;-----       HIGH RESOLUTION READ
7850                            <1> ;-----       GET VALUES FROM REGEN BUFFER AND CONVERT TO CODE POINT
7851                            <1>       ;MOV   DH,4               ; NUMBER OF PASSES
7852                            <1> S11:
7853 00002A48 8A06              <1>       MOV    AL, [eSI]          ; GET FIRST BYTE
7854 00002A4A 884500            <1>       MOV    [eBP], AL          ; SAVE IN STORAGE AREA
7855 00002A4D 45                <1>       INC    eBP                ; NEXT LOCATION
7856 00002A4E 8A8600200000      <1>       MOV    AL, [eSI+2000H]          ; GET LOWER REGION BYTE
7857 00002A54 884500            <1>       MOV    [eBP], AL          ; ADJUST AND STORE
7858 00002A57 45                <1>       INC    eBP
7859 00002A58 83C650            <1>       ADD    eSI, 80                  ; POINTER INTO REGEN
7860 00002A5B FECE              <1>       DEC    DH                 ; LOOP CONTROL
7861 00002A5D 75E9              <1>       JNZ    short S11          ; DO IT SOME MORE
7862 00002A5F EB1D              <1>       JMP    SHORT S14          ; GO MATCH THE SAVED CODE POINTS
7863                            <1>
7864                            <1> ;-----       MEDIUM RESOLUTION READ
7865                            <1> S12:
7866 00002A61 66D1E6            <1>       SAL    SI, 1              ; OFFSET*2 SINCE 2 BYTES/CHAR
7867                            <1>       ;MOV   DH, 4              ; NUMBER OF PASSES
7868                            <1> S13:
7869 00002A64 E84B000000        <1>       CALL   S23                ; GET BYTES FROM REGEN INTO SINGLE SAVE
7870 00002A69 81C6FE1F0000      <1>       ADD    eSI, 2000H-2       ; GO TO LOWER REGION
7871 00002A6F E840000000        <1>       CALL   S23                ; GET THIS PAIR INTO SAVE
7872 00002A74 81EEB21F0000      <1>       SUB    eSI, 2000H-80+2          ; ADJUST POINTER BACK INTO UPPER
7873 00002A7A FECE              <1>       DEC    DH
7874 00002A7C 75E6              <1>       JNZ    short S13          ; KEEP GOING UNTIL ALL 8 DONE
7875                            <1>
7876                            <1> ;-----       SAVE AREA HAS CHARACTER IN IT, MATCH IT
7877                            <1> S14:                            ; FIND_CHAR
7878 00002A7E BF[3C570100]      <1>       MOV    eDI, CRT_CHAR_GEN  ; ESTABLISH ADDRESSING
7879 00002A83 83ED08            <1>       SUB    eBP, 8             ; ADJUST POINTER TO START OF SAVE AREA
7880 00002A86 89EE              <1>       MOV    eSI, eBP
7881                            <1> S15:
7882 00002A88 66B80001          <1>       mov    ax, 256                  ; NUMBER TO TEST AGAINST
7883                            <1> S16:
7884 00002A8C 56                <1>       PUSH   eSI                ; SAVE SAVE AREA POINTER
7885 00002A8D 57                <1>       PUSH   eDI                ; SAVE CODE POINTER
7886                            <1>       ;MOV   eCX, 4             ; NUMBER OF WORDS TO MATCH
7887                            <1>       ;REPE  CMPSW              ; COMPARE THE 8 BYTES AS WORDS
7888 00002A8E A7                <1>       cmpsd                     ; compare first 4 bytes
7889 00002A8F 7501              <1>       jne    short S17          ;
7890 00002A91 A7                <1>       cmpsd                     ; compare last 4 bytes
7891                            <1> S17:
7892 00002A92 5F                <1>       POP    eDI                ; RECOVER THE POINTERS
7893 00002A93 5E                <1>       POP    eSI
7894                            <1>       ;JZ    short S18          ; IF ZERO FLAG SET, THEN MATCH OCCURRED
7895 00002A94 7407              <1>       je     short S18
7896                            <1>       ;                         ; NO MATCH, MOVE ON TO NEXT
7897 00002A96 83C708            <1>       ADD    eDI, 8             ; NEXT CODE POINT
7898 00002A99 6648              <1>       dec    ax                 ; LOOP CONTROL
7899 00002A9B 75EF              <1>       JNZ    short S16          ; DO ALL OF THEM
7900                            <1>
7901                            <1> ;-----       CHARACTER IS FOUND ( AL=0 IF NOT FOUND )
7902                            <1> S18:
7903 00002A9D 83C408            <1>       ADD    eSP, 8             ; READJUST THE STACK, THROW AWAY SAVE
7904 00002AA0 C3                <1>       retn                      ; ALL DONE
7905                            <1>
7906                            <1> ; 12/04/2021
7907                            <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
7908                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7909                            <1> ;----------------------------------------
7910                            <1> ; EXPAND BYTE
7911                            <1> ;  THIS ROUTINE TAKES THE BYTE IN AL AND DOUBLES ALL
7912                            <1> ;  OF THE BITS, TURNING THE 8 BITS INTO 16 BITS.
7913                            <1> ;  THE RESULT IS LEFT IN AX
7914                            <1> ;----------------------------------------
7915                            <1> S21:
7916                            <1>       ;PUSH  CX                 ; SAVE REGISTER
7917                            <1>       ; 12/04/2021
7918 00002AA1 51                <1>       push   ecx
7919                            <1>       ;MOV   CX, 8              ; SHIFT COUNT REGISTER FOR ONE BYTE
7920 00002AA2 B108              <1>       mov    cl, 8
7921                            <1> S22:
7922 00002AA4 D0C8              <1>       ROR    AL,1               ; SHIFT BITS, LOW BIT INTO CARRY FLAG
7923 00002AA6 66D1DD            <1>       RCR    BP,1               ; MOVE CARRY FLAG (LOW BIT INTO RESULTS
7924 00002AA9 66D1FD            <1>       SAR    BP,1               ; SIGN EXTEND HIGH BIT (DOUBLE IT)
7925                            <1>       ;LOOP  S22                ; REPEAT FOR ALL 8 BITS
7926 00002AAC FEC9              <1>       dec    cl
7927 00002AAE 75F4              <1>       jnz    short S22
7928 00002AB0 6695              <1>       XCHG   AX, BP             ; MOVE RESULTS TO PARAMETER REGISTER
7929                            <1>       ;POP   CX                 ; RECOVER REGISTER
7930                            <1>       ; 12/04/2021
7931 00002AB2 59                <1>       pop    ecx
7932 00002AB3 C3                <1>       RETn                      ; ALL DONE
```

```
                                <1>
                                <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
                                <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
                                <1> ;------------------------------------------------
                                <1> ; MED_READ_BYTE
                                <1> ; THIS ROUTINE WILL TAKE 2 BYTES FROM THE REGEN BUFFER,
                                <1> ;  COMPARE AGAINST THE CURRENT FOREGROUND COLOR, AND PLACE
                                <1> ;  THE CORRESPONDING ON/OFF BIT PATTERN INTO THE CURRENT
                                <1> ;  POSITION IN THE SAVE AREA
                                <1> ; ENTRY --
                                <1> ;  SI,DS = POINTER TO REGEN AREA OF INTEREST
                                <1> ;  BX = EXPANDED FOREGROUND COLOR
                                <1> ;  BP = POINTER TO SAVE AREA
                                <1> ; EXIT --
                                <1> ;  SI AND BP ARE INCREMENTED
                                <1> ;------------------------------------------------
                                <1> S23:
00002AB4 66AD                   <1>       LODSW                     ; GET FIRST BYTE AND SECOND BYTES
00002AB6 86C4                   <1>       XCHG   AL, AH             ; SWAP FOR COMPARE
00002AB8 66B900C0               <1>       MOV    CX, 0C000H         ; 2 BIT MASK TO TEST THE ENTRIES
00002ABC B200                   <1>       MOV    DL, 0              ; RESULT REGISTER
                                <1> S24:
00002ABE 6685C8                 <1>       TEST   AX, CX             ; IS THIS SECTION BACKCROUND?
00002AC1 7401                   <1>         JZ     short S25              ; IF ZERO, IT IS BACKGROUND (CARRY=0)
00002AC3 F9                     <1>       STC                       ; WASN'T, SO SET CARRY
                                <1> S25:
00002AC4 D0D2                   <1>       RCL    DL, 1              ; MOVE THAT BIT INTO THE RESULT
00002AC6 66C1E902               <1>       SHR    CX, 2              ; MOVE THE MASK TO THE RIGHT BY 2 BITS
00002ACA 73F2                   <1>       JNC    short S24          ; DO IT AGAIN IF MASK DIDN'T FALL OUT
00002ACC 885500                 <1>       MOV    [eBP], DL          ; STORE RESULT IN SAVE AREA
00002ACF 45                     <1>       INC    eBP                ; ADJUST POINTER
00002AD0 C3                     <1>       RETn                      ; ALL DONE
                                <1>
                                <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
                                <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
                                <1> ;----------------------------------------
                                <1> ; V4_POSITION
                                <1> ;  THIS ROUTINE TAKES THE CURSOR POSITION CONTAINED IN
                                <1> ;  THE MEMORY LOCATION, AND CONVERTS IT INTO AN OFFSET
                                <1> ;  INTO THE REGEN BUFFER, ASSUMING ONE BYTE/CHAR.
                                <1> ;  FOR MEDIUM RESOLUTION GRAPHICS, THE NUMBER MUST
                                <1> ;  BE DOUBLED.
                                <1> ; ENTRY -- NO REGISTERS,MEMORY LOCATION @CURSOR_POSN IS USED
                                <1> ; EXIT--
                                <1> ;  AX CONTAINS OFFSET INTO REGEN BUFFER
                                <1> ;----------------------------------------
                                <1> S26:
00002AD1 0FB705[C6800100]       <1>       movzx  eax, word [CURSOR_POSN]   ; GET CURRENT CURSOR
                                <1> GRAPH_POSN:
00002AD8 53                     <1>       PUSH   eBX                ; SAVE REGISTER
00002AD9 0FB6D8                 <1>       movzx  ebx, al                 ; SAVE A COPY OF CURRENT CURSOR
00002ADC A0[BC6A0000]           <1>       MOV    AL, [CRT_COLS]          ; GET BYTES PER COLUMN
00002AE1 F6E4                   <1>       MUL    AH                 ; MULTIPLY BY ROWS
00002AE3 66C1E002               <1>       SHL    AX, 2              ; MULTIPLY * 4 SINCE 4 ROWS/BYTE
00002AE7 01D8                   <1>       ADD    eAX, eBX           ; DETERMINE OFFSET
00002AE9 5B                     <1>       POP    eBX                ; RECOVER POINTER
00002AEA C3                     <1>       RETn                      ; ALL DONE
                                <1>
                                <1> ; 09/07/2016
                                <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
                                <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
                                <1> ;------------------------------------------
                                <1> ; SET_COLOR
                                <1> ;     THIS ROUTINE WILL ESTABLISH THE BACKGROUND COLOR, THE OVERSCAN COLOR,
                                <1> ;     AND THE FOREGROUND COLOR SET FOR MEDIUM RESOLUTION GRAPHICS
                                <1> ; INPUT
                                <1> ;     (BH) HAS COLOR ID
                                <1> ;           IF BH=0, THE BACKGROUND COLOR VALUE IS SET
                                <1> ;                 FROM THE LOW BITS OF BL (0-31)
                                <1> ;           IF BH=1, THE PALETTE SELECTION IS MADE
                                <1> ;                 BASED ON THE LOW BIT OF BL:
                                <1> ;                       0 = GREEN, RED, YELLOW FOR COLORS 1,2,3
                                <1> ;                       1 = BLUE, CYAN, MAGENTA FOR COLORS 1,2,3
                                <1> ;     (BL) HAS THE COLOR VALUE TO BE USED
                                <1> ; OUTPUT
                                <1> ;     THE COLOR SELECTION IS UPDATED
                                <1> ;-----------------------------------------------
                                <1> SET_COLOR:
00002AEB 803D[BA6A0000]07       <1>         cmp     byte [CRT_MODE], 7      ; 09/07/2016
00002AF2 0F87F1EFFFFF           <1>       ja     VIDEO_RETURN       ; nothing to do for VGA modes
                                <1>
                                <1>       ;MOV   DX, [ADDR_6845]           ; I/O PORT FOR PALETTE
                                <1>       ;mov   dx, 3D4h
                                <1>       ;ADD   DX,5               ; OVERSCAN PORT
00002AF8 66BAD903               <1>       mov    dx, 3D9h
00002AFC A0[BD6A0000]           <1>       MOV    AL, [CRT_PALETTE]  ; GET THE CURRENT PALETTE VALUE
00002B01 08FF                   <1>       OR     BH, BH             ; IS THIS COLOR 0?
00002B03 7512                   <1>       JNZ    short M20          ; OUTPUT COLOR 1
                                <1>
                                <1> ;-----      HANDLE COLOR 0 BY SETTING THE BACKGROUND COLOR
                                <1>
00002B05 24E0                   <1>       AND    AL, 0E0H           ; TURN OFF LOW 5 BITS OF CURRENT
00002B07 80E31F                 <1>       AND    BL, 01FH           ; TURN OFF HIGH 3 BITS OF INPUT VALUE
00002B0A 08D8                   <1>       OR     AL, BL             ; PUT VALUE INTO REGISTER
                                <1> M19:                           ; OUTPUT THE PALETTE
00002B0C EE                     <1>       OUT    DX, AL             ; OUTPUT COLOR SELECTION TO 3D9 PORT
00002B0D A2[BD6A0000]           <1>       MOV    [CRT_PALETTE], AL  ; SAVE THE COLOR VALUE
00002B12 E9D2EFFFFF             <1>       JMP    VIDEO_RETURN
                                <1>
                                <1> ;-----      HANDLE COLOR 1 BY SELECTING THE PALETTE TO BE USED
                                <1>
                                <1> M20:
00002B17 24DF                   <1>       AND    AL, 0DFH           ; TURN OFF PALETTE SELECT BIT
00002B19 D0EB                   <1>       SHR    BL, 1              ; TEST THE LOW ORDER BIT OF BL
00002B1B 73EF                   <1>       JNC    short M19          ; ALREADY DONE
```

```
8038 00002B1D 0C20              <1>       OR    AL, 20H                   ; TURN ON PALETTE SELECT BIT
8039 00002B1F EBEB              <1>       JMP   short M19           ; GO DO IT
8040                            <1>
8041                            <1> ; 09/07/2016
8042                            <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
8043                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
8044                            <1> ;--------------------------------------------
8045                            <1> ; READ DOT -- WRITE DOT
8046                            <1> ; THESE ROUTINES WILL WRITE A DOT, OR READ THE
8047                            <1> ;  DOT AT THE INDICATED LOCATION
8048                            <1> ; ENTRY --
8049                            <1> ;   DX = ROW (0-199)      (THE ACTUAL VALUE DEPENDS ON THE MODE)
8050                            <1> ;   CX = COLUMN ( 0-639) ( THE VALUES ARE NOT RANGE CHECKED )
8051                            <1> ;   AL = DOT VALUE TO WRITE (1,2 OR 4 BITS DEPENDING ON MODE,
8052                            <1> ;     REQUIRED FOR WRITE DOT ONLY, RIGHT JUSTIFIED)
8053                            <1> ;     BIT 7 OF AL = 1 INDICATES XOR THE VALUE INTO THE LOCATION
8054                            <1> ;   DS = DATA SEGMENT
8055                            <1> ;   ES = REGEN SEGMENT
8056                            <1> ;
8057                            <1> ; EXIT
8058                            <1> ;     AL = DOT VALUE READ, RIGHT JUSTIFIED, READ ONLY
8059                            <1> ;--------------------------------------------
8060                            <1>
8061                            <1> READ_DOT:
8062                            <1>       ; 09/07/2016
8063 00002B21 8A25[BA6A0000]    <1>       mov   ah,  [CRT_MODE]
8064 00002B27 80FC07            <1>       cmp   ah, 7 ; 6!?
8065 00002B2A 760A              <1>       jna   short read_dot_cga
8066                            <1>
8067 00002B2C E8D1030000        <1>       call  vga_read_pixel
8068                            <1>       ; al = pixel value
8069 00002B31 E9B8EFFFFF        <1>       jmp   _video_return
8070                            <1>
8071                            <1> read_dot_cga:
8072                            <1>       ;je   VIDEO_RETURN ; 7
8073 00002B36 80FC04            <1>       cmp   ah, 4 ; graphics ?
8074 00002B39 0F82AAEFFFFF      <1>       jb    VIDEO_RETURN ; no, text mode, nothing to do
8075                            <1>
8076 00002B3F E853000000        <1>       CALL  R3                ; DETERMINE BYTE POSITION OF DOT
8077 00002B44 8A06              <1>       MOV   AL, [eSI]         ; GET THE BYTE
8078 00002B46 20E0              <1>       AND   AL, AH            ; MASK OFF THE OTHER BITS IN THE BYTE
8079 00002B48 D2E0              <1>       SHL   AL, CL            ; LEFT JUSTIFY THE VALUE
8080 00002B4A 88F1              <1>       MOV   CL, DH            ; GET NUMBER OF BITS IN RESULT
8081 00002B4C D2C0              <1>       ROL   AL, CL            ; RIGHT JUSTIFY THE RESULT
8082                            <1>       ;JMP  VIDEO_RETURN      ; RETURN FROM VIDEO I/O
8083 00002B4E 0FB6C0            <1>       movzx eax, al
8084 00002B51 E998EFFFFF        <1>       jmp   _video_return
8085                            <1>
8086                            <1> ; 12/04/2021
8087                            <1> ; 09/07/2016
8088                            <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
8089                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
8090                            <1>
8091                            <1> WRITE_DOT:
8092                            <1>       ; 09/07/2016
8093 00002B56 8A25[BA6A0000]    <1>       mov   ah, [CRT_MODE]
8094 00002B5C 80FC07            <1>       cmp   ah, 7 ; 6!?
8095 00002B5F 760A              <1>       jna   short write_dot_cga
8096                            <1>
8097 00002B61 E80B030000        <1>       call  vga_write_pixel
8098 00002B66 E97EEFFFFF        <1>       jmp   VIDEO_RETURN
8099                            <1>
8100                            <1> write_dot_cga:
8101                            <1>       ;je   VIDEO_RETURN ; 7
8102 00002B6B 80FC04            <1>       cmp   ah, 4 ; graphics ?
8103 00002B6E 0F8275EFFFFF      <1>       jb    VIDEO_RETURN ; no, text mode, nothing to do
8104                            <1>
8105                            <1>       ;;PUSH AX               ; SAVE DOT VALUE
8106                            <1>       ;PUSH  AX               ; TWICE
8107                            <1>       ; 12/04/2021
8108 00002B74 50                <1>       push  eax
8109 00002B75 E81D000000        <1>       CALL  R3                ; DETERMINE BYTE POSITION OF THE DOT
8110 00002B7A D2E8              <1>       SHR   AL, CL            ; SHIFT TO SET UP THE BITS FOR OUTPUT
8111 00002B7C 20E0              <1>       AND   AL, AH            ; STRIP OFF THE OTHER BITS
8112 00002B7E 8A0E              <1>       MOV   CL, [eSI]         ; GET THE CURRENT BYTE
8113                            <1>       ;POP  BX                ; RECOVER XOR FLAG
8114                            <1>       ; 12/04/2021
8115 00002B80 5B                <1>       pop   ebx
8116 00002B81 F6C380            <1>       TEST  BL, 80H                ; IS IT ON
8117 00002B84 750D              <1>       JNZ   short R2          ; YES, XOR THE DOT
8118 00002B86 F6D4              <1>       NOT   AH                ; SET MASK TO REMOVE THE INDICATED BITS
8119 00002B88 20E1              <1>       AND   CL, AH
8120 00002B8A 08C8              <1>       OR    AL, CL            ; OR IN THE NEW VALUE OF THOSE BITS
8121                            <1> R1:                           ; FINISH_DOT
8122 00002B8C 8806              <1>       MOV   [eSI], AL         ; RESTORE THE BYTE IN MEMORY
8123                            <1>       ;;POP  AX
8124 00002B8E E956EFFFFF        <1>       JMP   VIDEO_RETURN      ; RETURN FROM VIDEO I/O
8125                            <1> R2:                           ; XOR_DOT
8126 00002B93 30C8              <1>       XOR   AL, CL            ; EXCLUSIVE OR THE DOTS
8127 00002B95 EBF5              <1>       JMP   short R1          ; FINISH UP THE WRITING
8128                            <1>
8129                            <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
8130                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
8131                            <1>
8132                            <1> ;--------------------------------------------
8133                            <1> ; THIS SUBROUTINE DETERMINES THE REGEN BYTE LOCATION OF THE
8134                            <1> ; INDICATED ROW COLUMN VALUE IN GRAPHICS MODE.
8135                            <1> ; ENTRY --
8136                            <1> ;  DX = ROW VALUE (0-199)
8137                            <1> ;  CX = COLUMN VALUE (0-639)
8138                            <1> ; EXIT --
8139                            <1> ;  SI = OFFSET INTO REGEN BUFFER FOR BYTE OF INTEREST
8140                            <1> ;  AH = MASK TO STRIP OFF THE BITS OF INTEREST
8141                            <1> ;  CL = BITS TO SHIFT TO RIGHT JUSTIFY THE MASK IN AH
8142                            <1> ;  DH = # BITS IN RESULT
```

```
8143                            <1> ;  BX = MODIFIED
8144                            <1> ;-------------------------------------------------
8145                            <1> R3:
8146                            <1>
8147                            <1> ;-----      DETERMINE 1ST BYTE IN INDICATED ROW BY MULTIPLYING ROW VALUE BY 40
8148                            <1> ;-----       ( LOW BIT OF ROW DETERMINES EVEN/ODD, 80 BYTES/ROW )
8149                            <1>
8150 00002B97 0FB7F0           <1>       movzx  esi, ax                  ; WILL SAVE AL AND AH DURING OPERATION
8151 00002B9A B028             <1>       MOV    AL, 40
8152 00002B9C F6E2             <1>       MUL    DL                ; AX= ADDRESS OF START OF INDICATED ROW
8153 00002B9E A808             <1>       TEST   AL, 08H           ; TEST FOR EVEN/ODD ROW CALCULATED
8154 00002BA0 7404             <1>       JZ     short R4          ; JUMP IF EVEN ROW
8155 00002BA2 6605D81F         <1>       ADD    AX, 2000H-40      ; OFFSET TO LOCATION OF ODD ROWS ADJUST
8156                            <1> R4:                          ; EVEN_ROW
8157 00002BA6 6696             <1>       XCHG   SI, AX            ; MOVE POINTER TO (SI) AND RECOVER (AX)
8158 00002BA8 81C600800B00     <1>       add    esi, 0B8000h
8159 00002BAE 6689CA           <1>       MOV    DX, CX            ; COLUMN VALUE TO DX
8160                            <1>
8161                            <1> ;-----      DETERMINE GRAPHICS MODE CURRENTLY IN EFFECT
8162                            <1>
8163                            <1> ; SET UP THE REGISTERS ACCORDING TO THE MODE
8164                            <1> ; CH = MASK FOR LOW OF COLUMN ADDRESS ( 7/3 FOR HIGH/MED RES )
8165                            <1> ; CL = # OF ADDRESS BITS IN COLUMN VALUE ( 3/2 FOR H/M )
8166                            <1> ; BL = MASK TO SELECT BITS FROM POINTED BYTE ( 80H/C0H FOR H/M )
8167                            <1> ; BH = NUMBER OF VALID BITS IN POINTED BYTE ( 1/2 FOR H/M )
8168                            <1>
8169 00002BB1 66BBC002         <1>       MOV    BX, 2C0H
8170 00002BB5 66B90203         <1>       MOV    CX, 302H          ; SET PARMS FOR MED RES
8171 00002BB9 803D[BA6A0000]06 <1>       CMP    byte [CRT_MODE], 6
8172 00002BC0 7208             <1>       JC     short R5          ; HANDLE IF MED RES
8173 00002BC2 66BB8001         <1>       MOV    BX, 180H
8174 00002BC6 66B90307         <1>       MOV    CX, 703H          ; SET PARMS FOR HIGH RES
8175                            <1>
8176                            <1> ;-----      DETERMINE BIT OFFSET IN BYTE FROM COLUMN MASK
8177                            <1> R5:
8178 00002BCA 20D5             <1>       AND    CH, DL            ; ADDRESS OF PEL WITHIN BYTE TO CH
8179                            <1>
8180                            <1> ;-----      DETERMINE BYTE OFFSET FOR THIS LOCATION IN COLUMN
8181                            <1>
8182 00002BCC 66D3EA           <1>       SHR    DX, CL            ; SHIFT BY CORRECT AMOUNT
8183 00002BCF 6601D6           <1>       ADD    SI, DX            ; INCREMENT THE POINTER
8184 00002BD2 88FE             <1>       MOV    DH, BH            ; GET THE # OF BITS IN RESULT TO DH
8185                            <1>
8186                            <1> ;-----      MULTIPLY BH (VALID BITS IN BYTE) BY CH (BIT OFFSET)
8187                            <1>
8188 00002BD4 28C9             <1>       SUB    CL, CL            ; ZERO INTO STORAGE LOCATION
8189                            <1> R6:
8190 00002BD6 D0C8             <1>       ROR    AL, 1             ; LEFT JUSTIFY VALUE IN AL (FOR WRITE)
8191 00002BD8 00E9             <1>       ADD    CL, CH            ; ADD IN THE BIT OFFSET VALUE
8192 00002BDA FECF             <1>       DEC    BH                ; LOOP CONTROL
8193 00002BDC 75F8             <1>       JNZ    short R6          ; ON EXIT, CL HAS COUNT TO RESTORE BITS
8194 00002BDE 88DC             <1>       MOV    AH, BL            ;  GET MASK TO AH
8195 00002BE0 D2EC             <1>       SHR    AH, CL            ;  MOVE THE MASK TO CORRECT LOCATION
8196 00002BE2 C3               <1>       RETn                     ;  RETURN WITH EVERYTHING SET UP
8197                            <1>
8198                            <1> load_dac_palette:
8199                            <1>       ; 29/07/2016
8200                            <1>       ; 23/07/2016
8201                            <1>       ; 03/07/2016 (TRDOS 386 = TRDOS v2.0)
8202                            <1>       ; (set_mode_vga)
8203                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
8204                            <1>       ; vgabios-0.7a (2011)
8205                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
8206                            <1>       ; 'vgabios.c', 'load_dac_palette'
8207                            <1>       ;
8208                            <1>       ; Oracle VirtualBox 5.0.24 VGABios Source Code
8209                            <1>       ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
8210                            <1>       ;
8211                            <1>       ; INPUT -> AH = DAC selection number (3, 2 or 1)
8212                            <1>       ; OUTPUT -> ECX = 0, AX = 0
8213                            <1>       ; (Modifed registers: EAX, ECX, EDX, ESI)
8214                            <1>       ;
8215 00002BE3 66BAC803         <1>       mov    dx, 3C8h  ; VGAREG_DAC_WRITE_ADDRESS
8216 00002BE7 28C0             <1>       sub    al, al ; 0
8217 00002BE9 EE               <1>       out    dx, al ; 0 ; color index, always 0 at the beginning
8218 00002BEA 6642             <1>       inc    dx   ; 3C9h ; VGAREG_DAC_DATA
8219 00002BEC B900010000       <1>       mov    ecx, 256   ; always 256*3 values
8220                            <1>       ;push  esi
8221 00002BF1 88E0             <1>       mov    al, ah
8222 00002BF3 B43F             <1>       mov    ah, 3Fh      ; 3Fh except DAC selection number 3
8223 00002BF5 3C02             <1>       cmp    al, 2
8224 00002BF7 7414             <1>       je     short l_dac_p_2
8225 00002BF9 7719             <1>       ja     short l_dac_p_3
8226 00002BFB 20C0             <1>       and    al, al
8227 00002BFD 7507             <1>       jnz    short l_dac_p_1
8228                            <1> l_dac_p_0:
8229 00002BFF BE[FC510100]     <1>       mov    esi, palette0
8230 00002C04 EB15             <1>       jmp    short l_dac_p_4
8231                            <1> l_dac_p_1:
8232 00002C06 BE[BC520100]     <1>       mov    esi, palette1
8233 00002C0B EB0E             <1>       jmp    short l_dac_p_4
8234                            <1> l_dac_p_2:
8235 00002C0D BE[7C530100]     <1>       mov    esi, palette2
8236 00002C12 EB07             <1>       jmp    short l_dac_p_4
8237                            <1> l_dac_p_3:
8238 00002C14 B4FF             <1>       mov    ah, 0FFh ; dac registers
8239 00002C16 BE[3C540100]     <1>       mov    esi, palette3
8240                            <1> l_dac_p_4:
8241 00002C1B AC               <1>       lodsb
8242 00002C1C EE               <1>       out    dx, al  ; Red
8243 00002C1D AC               <1>       lodsb
8244 00002C1E EE               <1>       out    dx, al ; Green
8245 00002C1F AC               <1>       lodsb
8246 00002C20 EE               <1>       out    dx, al ; Blue
8247 00002C21 20E4             <1>       and    ah, ah
```

```
8248 00002C23 7405              <1>       jz    short l_dac_p_5
8249 00002C25 FECC              <1>       dec   ah
8250 00002C27 E2F2              <1>       loop  l_dac_p_4
8251                            <1>       ;pop  esi
8252 00002C29 C3                <1>       retn
8253                            <1> l_dac_p_5:
8254                            <1>       ; 29/07/2016
8255 00002C2A FEC9              <1>       dec   cl
8256 00002C2C 7407              <1>       jz    short l_dac_p_7
8257                            <1>       ;
8258 00002C2E 28C0              <1>       sub   al, al ; 0
8259                            <1> l_dac_p_6:
8260 00002C30 EE                <1>       out   dx, al ; outb(VGAREG_DAC_DATA,0);
8261 00002C31 EE                <1>       out   dx, al
8262 00002C32 EE                <1>       out   dx, al
8263 00002C33 E2FB              <1>       loop  l_dac_p_6
8264                            <1> l_dac_p_7:
8265                            <1>       ;pop  esi
8266 00002C35 C3                <1>       retn
8267                            <1>
8268                            <1> gray_scale_summing:
8269                            <1>       ; 12/04/2021
8270                            <1>       ; 03/07/2016 (TRDOS 386 = TRDOS v2.0)
8271                            <1>       ; (set_mode_vga)
8272                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
8273                            <1>       ; vgabios-0.7a (2011)
8274                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
8275                            <1>       ; 'vgabios.c', 'biosfn_perform_gray_scale_summing'
8276                            <1>       ;
8277                            <1>       ; Oracle VirtualBox 5.0.24 VGABios Source Code
8278                            <1>       ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
8279                            <1>       ;
8280                            <1>
8281                            <1>       ; INPUT -> EBX = Start address (color index <= 255)
8282                            <1>       ;         ECX = Count (<= 256)
8283                            <1>       ; OUTPUT -> (E)CX = 0
8284                            <1>       ; (Modifed registers: EAX, ECX, EDX, EBX)
8285                            <1>
8286 00002C36 66BADA03          <1>       mov   dx, 3DAh ; VGAREG_ACTL_RESET
8287 00002C3A EC                <1>       in    al, dx
8288 00002C3B 30C0              <1>       xor   al, al ; 0
8289 00002C3D 66BAC003          <1>       mov   dx, 3C0h ; VGAREG_ACTL_ADDRESS
8290 00002C41 EE                <1>       out   dx, al ; clear bit 5
8291                            <1>                    ; (while loading palette registers)
8292                            <1>       ; set read address and switch to read mode
8293                            <1> g_s_s_1:
8294 00002C42 66BAC703          <1>       mov   dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
8295 00002C46 88D8              <1>       mov   al, bl
8296 00002C48 EE                <1>       out   dx, al
8297                            <1>       ; get 6-bit wide RGB data values
8298                            <1>       ; intensity = (0.3*Red)+(0.59*Green)+(0.11*Blue)
8299                            <1>       ; i = ( ( 77*r + 151*g + 28*b ) + 0x80 ) >> 8;
8300 00002C49 66BAC903          <1>       mov   dx, 3C9h ; VGAREG_DAC_DATA
8301 00002C4D EC                <1>       in    al, dx ; red
8302 00002C4E B44D              <1>       mov   ah, 77 ; 0.3* Red
8303 00002C50 F6E4              <1>       mul   ah
8304                            <1>       ;push ax
8305                            <1>       ; 12/04/2021
8306 00002C52 50                <1>       push  eax
8307 00002C53 EC                <1>       in    al, dx ; green
8308 00002C54 B497              <1>       mov   ah, 151  ; 0.59 * Green
8309 00002C56 F6E4              <1>       mul   ah
8310                            <1>       ;push ax
8311                            <1>       ; 12/04/2021
8312 00002C58 50                <1>       push  eax
8313 00002C59 EC                <1>       in    al, dx ; blue
8314 00002C5A B41C              <1>       mov   ah, 28 ; 0.11 * Blue
8315 00002C5C F6E4              <1>       mul   ah
8316                            <1>       ;pop  dx
8317                            <1>       ; 12/04/2021
8318 00002C5E 5A                <1>       pop   edx
8319 00002C5F 6601D0            <1>       add   ax, dx
8320                            <1>       ;pop  dx
8321                            <1>       ; 12/04/2021
8322 00002C62 5A                <1>       pop   edx
8323 00002C63 6601D0            <1>       add   ax, dx
8324 00002C66 66058000          <1>       add   ax, 80h
8325 00002C6A B03F              <1>       mov   al, 3Fh
8326 00002C6C 38C4              <1>       cmp   ah, al ; if(i>0x3f)i=0x3f
8327 00002C6E 7602              <1>       jna   short g_s_s_2
8328 00002C70 88C4              <1>       mov   ah, al
8329                            <1> g_s_s_2:
8330 00002C72 66BAC803          <1>       mov   dx, 3C8h  ; VGAREG_DAC_WRITE_ADDRESS
8331 00002C76 88D8              <1>       mov   al, bl ; color index
8332 00002C78 EE                <1>       out   dx, al
8333 00002C79 88E0              <1>       mov   al, ah ; intensity
8334 00002C7B 6642              <1>       inc   dx ; 3C9h ; VGAREG_DAC_DATA
8335 00002C7D EE                <1>       out   dx, al ; R (R=G=B)
8336 00002C7E 88E0              <1>       mov   al, ah ; intensity
8337 00002C80 EE                <1>       out   dx, al ; G (R=G=B)
8338 00002C81 88E0              <1>       mov   al, ah ; intensity
8339 00002C83 EE                <1>       out   dx, al ; B (R=G=B)
8340 00002C84 6649              <1>       dec   cx
8341 00002C86 7404              <1>       jz    short g_s_s_3
8342 00002C88 FEC3              <1>       inc   bl    ; next color index value
8343 00002C8A EBB6              <1>       jmp   short g_s_s_1
8344                            <1> g_s_s_3:
8345 00002C8C 66BADA03          <1>       mov   dx, 3DAh ; VGAREG_ACTL_RESET
8346 00002C90 EC                <1>       in    al, dx
8347 00002C91 B020              <1>       mov   al, 20h
8348 00002C93 66BAC003          <1>       mov   dx, 3C0h ; VGAREG_ACTL_ADDRESS
8349 00002C97 EE                <1>       out   dx, al ; 20h -> set bit 5
8350                            <1>                      ; (after loading palette regs)
8351 00002C98 C3                <1>       retn
8352                            <1>
```

```
                                <1> vga_write_char_attr:
                                <1> vga_write_char_only:
                                <1>       ; 08/07/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ;
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
                                <1>       ; 'vgabios.c', 'biosfn_write_char_attr'
                                <1>       ; 'biosfn_write_char_only'
                                <1>
                                <1>       ; INPUT ->
                                <1>       ; [CRT_MODE] = current video mode (>7)
                                <1>       ; CX = Count of characters to write
                                <1>       ; AL = Character to write
                                <1>       ; BL = Color of character
                                <1>       ; OUTPUT ->
                                <1>       ; Regen buffer updated
                                <1>
00002C99 8A25[BA6A0000]         <1>       mov    ah, [CRT_MODE]
00002C9F 668B15[C6800100]       <1>       mov    dx, [CURSOR_POSN] ; cursor pos for page 0
                                <1>
00002CA6 BE[D66A0000]           <1>       mov    esi, vga_modes
00002CAB 89F7                   <1>       mov    edi, esi
00002CAD 83C710                 <1>       add    edi, vga_mode_count
                                <1> vga_wca_0:
00002CB0 AC                     <1>       lodsb
00002CB1 38E0                   <1>       cmp    al, ah ; [CRT_MODE]
00002CB3 7405                   <1>       je     short vga_wca_2
00002CB5 39FE                   <1>       cmp    esi, edi
00002CB7 72F7                   <1>       jb     short vga_wca_0
                                <1> vga_wca_1:
00002CB9 C3                     <1>       retn   ; nothing to do
                                <1> vga_wca_2:
00002CBA 83C64F                 <1>       add    esi, vga_memmodel - (vga_modes + 1)
                                <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
                                <1>
                                <1>       ; biosfn_write_char_attr (car,page,attr,count)
                                <1>       ; AL = car, page = 0, BL = attr, CX = count
00002CBD 803E04                 <1>       cmp    byte [esi], PLANAR4
00002CC0 741D                   <1>       je     short vga_wca_planar
00002CC2 803E03                 <1>       cmp    byte [esi], PLANAR1
00002CC5 7418                   <1>       je     short vga_wca_planar
                                <1> vga_wca_linear8:
                                <1>       ; while((count-->0) && (xcurs<nbcols))
                                <1>       ; CX = count
00002CC7 6621C9                 <1>       and    cx, cx
00002CCA 74ED                   <1>       jz     short vga_wca_1
00002CCC 3A15[BC6A0000]         <1>       cmp    dl, [CRT_COLS]
00002CD2 73E5                   <1>       jnb    short vga_wca_1
                                <1>       ; write_gfx_char_lin(car,attr,xcurs,ycurs,nbcols);
                                <1>       ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
                                <1>       ; [CRT_COLS] = nbcols
00002CD4 E81E000000             <1>       call   write_gfx_char_lin
00002CD9 6649                   <1>       dec    cx ; count
00002CDB FEC2                   <1>       inc    dl ; xcurs
00002CDD EBE8                   <1>       jmp    short vga_wca_linear8
                                <1> vga_wca_planar:
                                <1>       ; while((count-->0) && (xcurs<nbcols))
                                <1>       ; CX = count
00002CDF 6621C9                 <1>       and    cx, cx
00002CE2 74D5                   <1>       jz     short vga_wca_1
00002CE4 3A15[BC6A0000]         <1>       cmp    dl, [CRT_COLS]
00002CEA 73CD                   <1>       jnb    short vga_wca_1
                                <1>       ; write_gfx_char_pl4(car,attr,xcurs,ycurs,nbcols,cheight);
                                <1>       ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
                                <1>       ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
00002CEC E8A9000000             <1>       call   write_gfx_char_pl4
00002CF1 6649                   <1>       dec    cx ; count
00002CF3 FEC2                   <1>       inc    dl ; xcurs
00002CF5 EBE8                   <1>       jmp    short vga_wca_planar
                                <1>
                                <1> write_gfx_char_lin:
                                <1>       ; 08/01/2021
                                <1>       ; 05/01/2021 (TRDOS 386 v2.0.3)
                                <1>       ; 08/08/2016
                                <1>       ; 31/07/2016
                                <1>       ; 08/07/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ;
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
                                <1>       ; 'vgabios.c', 'write_gfx_char_lin'
                                <1>
                                <1>       ; write_gfx_char_lin(car,attr,xcurs,ycurs,nbcols)
                                <1>       ; INPUT ->
                                <1>       ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
                                <1>       ; [CRT_COLS] = nbcols
                                <1>       ; OUTPUT ->
                                <1>       ; Regen buffer updated
                                <1>
00002CF7 51                     <1>       push   ecx
00002CF8 53                     <1>       push   ebx
00002CF9 52                     <1>       push   edx
00002CFA 50                     <1>       push   eax
                                <1>       ; addr=xcurs*8+ycurs*nbcols*64;
                                <1>       ; 08/08/2016
00002CFB 0FB6F0                 <1>       movzx  esi, al ; car
                                <1>       ; 08/01/2021
                                <1>       ;movzx eax, dh ; ycurs
                                <1>       ;mov   ah, [CRT_COLS] ; nbcols
                                <1>       ;mul   ah
00002CFE A0[BC6A0000]           <1>       mov    al, [CRT_COLS]
00002D03 F6E6                   <1>       mul    dh
                                <1>       ;shl   ax, 6 ; * 64
00002D05 66C1E003               <1>       shl    ax, 3 ; 8 * ycurs * [CRT_COLS]
```

```
8458                                    <1>       ;sub   dh, dh
8459                                    <1>       ;shl   dx, 3 ; xcurs * 8
8460                                    <1>       ;movzx edi, dx
8461 00002D09 BF00000A00               <1>       mov    edi, 0A0000h
8462 00002D0E 30F6                     <1>       xor    dh, dh
8463 00002D10 6689D7                   <1>       mov    di, dx
8464                                    <1>       ;movzx edi, dl
8465 00002D13 66C1E703                 <1>       shl    di, 3 ; xcurs * 8
8466                                    <1>       ;xor   dh, dh
8467 00002D17 8A15[BE6A0000]           <1>       mov    dl, [CHAR_HEIGHT]
8468 00002D1D 66F7E2                   <1>       mul    dx
8469                                    <1>       ; eax = ycurs*nbcols*8*[CHAR_HEIGHT]
8470                                    <1>       ;add   edi, eax ; addr
8471                                    <1>       ;add   edi, 0A0000h
8472 00002D20 6601C7                   <1>       add    di, ax
8473                                    <1>       ;shl   si, 3 ; car * 8
8474 00002D23 30E4                     <1>       xor    ah, ah
8475 00002D25 A0[BE6A0000]             <1>       mov    al, [CHAR_HEIGHT]
8476 00002D2A 66F7E6                   <1>       mul    si
8477 00002D2D 6689C6                   <1>       mov    si, ax
8478                                    <1>       ;; esi = src = car * 8
8479                                    <1>       ; esi = src = car * [CHAR_HEIGHT]
8480                                    <1>       ; i = 0
8481                                    <1>       ;add   esi, vgafont8 ; fdata [src+i]
8482                                    <1>       ; 08/08/2016
8483 00002D30 A1[528D0100]             <1>       mov    eax, [VGA_INT43H]
8484 00002D35 09C0                     <1>       or     eax, eax ; 0 ?
8485 00002D37 743F                     <1>       jz     short wfxl_7 ; yes, default font
8486                                    <1>       ;cmp   eax, vgafont16
8487                                    <1>         ;je  short wgfxl_0
8488                                    <1>       ;cmp   eax, vgafont14
8489                                    <1>       ;je    short wgfxl_0
8490                                    <1>       ;cmp   eax, vgafont8
8491                                    <1>       ;je    short wgfxl_0
8492                                    <1>       ;; 05/01/2021 (TRDOS 386 v2.0.3)
8493                                    <1>       ;; user font
8494                                    <1>       ;mov   eax, VGAFONTUSR ; 8x16 or 8x8 or 8x14 font
8495                                    <1>       ;                      ; (256 characters)
8496                                    <1> wgfxl_0:
8497 00002D39 01C6                     <1>       add    esi, eax
8498                                    <1> wgfxl_1:
8499 00002D3B 28FF                     <1>       sub    bh, bh ; i = 0
8500                                    <1> wgfxl_2:
8501                                    <1>       ; for(i=0;i<8;i++)
8502 00002D3D 57                       <1>       push   edi ; addr
8503 00002D3E 0FB605[BC6A0000]         <1>       movzx  eax, byte [CRT_COLS] ; nbcols
8504 00002D45 F6E7                     <1>       mul    bh ; nbcols*i
8505 00002D47 66C1E003                 <1>       shl    ax, 3 ; i*nbcols*8
8506                                    <1>       ; dest=addr+i*nbcols*8;
8507 00002D4B 01C7                     <1>       add    edi, eax ; dest + j ; j = 0
8508 00002D4D B180                     <1>       mov    cl, 80h ; mask = 0x80;
8509                                    <1>       ; esi = fdata + src + i
8510                                    <1>       ; for(j=0;j<8;j++)
8511 00002D4F 29D2                     <1>       sub    edx, edx ; j = 0
8512                                    <1> wgfxl_3:
8513 00002D51 8A06                     <1>       mov    al, [esi] ; al = fdata[src+i]
8514 00002D53 20C8                     <1>       and    al, cl ; if (fdata[src+i] & mask)
8515 00002D55 7402                     <1>       jz     short wgfxl_4  ; data = 0, zf = 1
8516 00002D57 88D8                     <1>       mov    al, bl ; data = attr;
8517                                    <1> wgfxl_4:
8518                                    <1>       ; write_byte(0xa000,dest+j,data);
8519 00002D59 AA                       <1>       stosb  ; dest + j (+ 0A0000h)
8520                                    <1>       ;inc   dl ; j++
8521                                    <1>       ;cmp   dl, 8
8522 00002D5A 80FA07                   <1>       cmp    dl, 7
8523 00002D5D 720E                     <1>       jb     short wgfxl_5
8524 00002D5F 5F                       <1>       pop    edi
8525                                    <1>       ; 08/08/2016
8526                                    <1>       ;cmp   bh, 7
8527                                    <1>       ;jnb   short wgfxl_6
8528 00002D60 FEC7                     <1>       inc    bh ; i++
8529 00002D62 3A3D[BE6A0000]           <1>       cmp    bh, [CHAR_HEIGHT]
8530 00002D68 7309                     <1>       jnb    short wgfxl_6
8531 00002D6A 46                       <1>       inc    esi
8532 00002D6B EBD0                     <1>       jmp    short wgfxl_2
8533                                    <1> wgfxl_5:
8534 00002D6D D0E9                     <1>       shr    cl, 1 ; mask >>= 1;
8535 00002D6F FEC2                     <1>       inc    dl ; j++
8536 00002D71 EBDE                     <1>         jmp     short wgfxl_3
8537                                    <1> wgfxl_6:
8538 00002D73 58                       <1>       pop    eax
8539 00002D74 5A                       <1>       pop    edx
8540 00002D75 5B                       <1>       pop    ebx
8541 00002D76 59                       <1>       pop    ecx
8542 00002D77 C3                       <1>       retn
8543                                    <1> wfxl_7:
8544                                    <1>       ; 08/01/2021
8545                                    <1>       ; 05/01/2021
8546                                    <1>       ; Default font (8x8 or 8x14 or 8x16)
8547 00002D78 A0[BE6A0000]             <1>       mov    al, [CHAR_HEIGHT]
8548 00002D7D 3C08                     <1>       cmp    al, 8
8549 00002D7F 7507                     <1>       jne    short wfxl_8
8550 00002D81 B8[3C570100]             <1>       mov    eax, vgafont8
8551 00002D86 EBB1                     <1>       jmp    short wgfxl_0
8552                                    <1> wfxl_8:
8553 00002D88 3C0E                     <1>       cmp    al, 14
8554 00002D8A 7507                     <1>       jne    short wfxl_9
8555 00002D8C B8[3C5F0100]             <1>       mov    eax, vgafont14
8556 00002D91 EBA6                     <1>       jmp    short wgfxl_0
8557                                    <1> wfxl_9:
8558 00002D93 B8[3C6D0100]             <1>       mov    eax, vgafont16
8559 00002D98 EB9F                     <1>       jmp    short wgfxl_0
8560                                    <1>
8561                                    <1> write_gfx_char_pl4:
8562                                    <1>       ; 08/08/2016
```

```
                        <1>       ; 08/07/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>       ;
                        <1>       ; derived from 'Plex86/Bochs VGABios' source code
                        <1>       ; vgabios-0.7a (2011)
                        <1>       ; by the LGPL VGABios developers Team (2001-2008)
                        <1>       ; 'vgabios.c', 'write_gfx_char_pl4'
                        <1>
                        <1>       ; write_gfx_char_pl4(car,attr,xcurs,ycurs,nbcols,cheight)
                        <1>       ; INPUT ->
                        <1>       ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
                        <1>       ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
                        <1>       ; OUTPUT ->
                        <1>       ; Regen buffer updated
                        <1>
00002D9A 51             <1>       push   ecx
00002D9B 53             <1>       push   ebx
00002D9C 52             <1>       push   edx
00002D9D 50             <1>       push   eax
                        <1> wgfxpl_f0:
                        <1>       ; switch(cheight)
00002D9E 8A25[BE6A0000] <1>       mov    ah, [CHAR_HEIGHT]
00002DA4 80FC10         <1>       cmp    ah, 16 ; case 16:
00002DA7 7507           <1>       jne    short wgfxpl_f1
                        <1>       ; fdata = &vgafont16;
00002DA9 BE[3C6D0100]   <1>       mov    esi, vgafont16
00002DAE EB13           <1>       jmp    short wgfxpl_f3
                        <1> wgfxpl_f1:
00002DB0 80FC0E         <1>       cmp    ah, 14 ; case 14:
00002DB3 7507           <1>       jne    short wgfxpl_f2
00002DB5 BE[3C5F0100]   <1>       mov    esi, vgafont14
00002DBA EB07           <1>       jmp    short wgfxpl_f3
                        <1> wgfxpl_f2:
                        <1>       ; default:
                        <1>       ;  fdata = &vgafont8;
00002DBC BE[3C570100]   <1>       mov    esi, vgafont8
00002DC1 B408           <1>       mov    ah, 8
                        <1> wgfxpl_f3:
                        <1>       ; al = car
00002DC3 F6E4           <1>       mul    ah ; ah = cheight
00002DC5 25FFFF0000     <1>       and    eax, 0FFFFh ; car * cheight
                        <1>       ; src = car * cheight;
00002DCA 01C6           <1>       add    esi, eax ; esi = fdata[src+i]
                        <1>       ; addr=xcurs*8+ycurs*nbcols*64;
00002DCC 88F0           <1>       mov    al, dh ; ycurs
00002DCE 8A25[BC6A0000] <1>       mov    ah, [CRT_COLS] ; nbcols
00002DD4 F6E4           <1>       mul    ah
                        <1>       ; 08/08/2016
                        <1>       ;shl   ax, 6 ; * 64
00002DD6 66C1E003       <1>       shl    ax, 3 ; * 8
                        <1>       ;sub   dh, dh ; 0
                        <1>       ;shl   dx, 3 ; xcurs * 8
                        <1>       ;movzx edi, dx
00002DDA 0FB6FA         <1>       movzx  edi, dl
00002DDD 66C1E703       <1>       shl    di, 3 ; xcurs * 8
00002DE1 30F6           <1>       xor    dh, dh
00002DE3 8A15[BE6A0000] <1>       mov    dl, [CHAR_HEIGHT]
00002DE9 66F7E2         <1>       mul    dx
                        <1>       ; eax = ycurs*nbcols*8*[CHAR_HEIGHT]
00002DEC 01C7           <1>       add    edi, eax ; addr
00002DEE 81C700000A00   <1>       add    edi, 0A0000h
                        <1>       ;
                        <1>       ; outw(VGAREG_SEQU_ADDRESS, 0x0f02);
                        <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0205);
00002DF4 66BAC403       <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
00002DF8 66B8020F       <1>       mov    ax, 0F02h
00002DFC 66EF           <1>       out    dx, ax
00002DFE 66BACE03       <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
00002E02 66B80502       <1>       mov    ax, 0205h
00002E06 66EF           <1>       out    dx, ax
                        <1>       ;
00002E08 66BACE03       <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
00002E0C F6C380         <1>       test   bl, 80h ; if(attr&0x80)
00002E0F 7406           <1>       jz     short wgfxpl_f4 ; else
                        <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x1803);
00002E11 66B80318       <1>       mov    ax, 1803h
00002E15 EB04           <1>       jmp    short wgfxpl_f5
                        <1> wgfxpl_f4:
                        <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0003);
00002E17 66B80300       <1>       mov    ax, 0003h
                        <1> wgfxpl_f5:
00002E1B 66EF           <1>       out    dx, ax
                        <1>       ;
00002E1D 28FF           <1>       sub    bh, bh ; i = 0
                        <1> wgfxpl_0:
                        <1>       ; for(i=0;i<cheight;i++)
00002E1F 57             <1>       push   edi ; addr
00002E20 0FB605[BC6A0000] <1>     movzx  eax, byte [CRT_COLS] ; nbcols
00002E27 F6E7           <1>       mul    bh ; nbcols*i
                        <1>       ; dest=addr+i*nbcols
00002E29 01C7           <1>       add    edi, eax ; dest
00002E2B B580           <1>       mov    ch, 80h ; mask = 0x80;
                        <1>       ; for(j=0;j<8;j++)
00002E2D 28C9           <1>       sub    cl, cl ; j = 0
                        <1> wgfxpl_1:
00002E2F D2ED           <1>       shr    ch, cl ; mask=0x80>>j;
                        <1>       ;
                        <1>       ; outw(VGAREG_GRDC_ADDRESS, (mask << 8) | 0x08);
                        <1>       ; read_byte(0xa000,dest);
                        <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
00002E31 88EC           <1>       mov    ah, ch
00002E33 B008           <1>       mov    al, 8
00002E35 66EF           <1>       out    dx, ax
00002E37 8A07           <1>       mov    al, [edi] ; ? (io delay?)
                        <1>       ;
00002E39 28C0           <1>       sub    al, al ; attr = 0
```

```
8668                                    <1>       ; if (fdata[src+i] & mask)
8669 00002E3B 842E                     <1>       test  byte [esi], ch
8670 00002E3D 7404                     <1>       jz    short wgfxpl_2  ; zf = 1
8671                                    <1>       ; write_byte(0xa000,dest,attr&0x0f);
8672 00002E3F 88D8                     <1>       mov   al, bl ; attr;
8673 00002E41 240F                     <1>       and   al, 0Fh      ; attr&0x0f
8674                                    <1> wgfxpl_2:
8675                                    <1>       ; write_byte(0xa000,dest,0x00);
8676 00002E43 8807                     <1>       mov   [edi], al ; dest (+ 0A0000h)
8677 00002E45 FEC1                     <1>       inc   cl ; j++
8678 00002E47 80F908                   <1>       cmp   cl, 8
8679 00002E4A 72E3                     <1>       jb    short wgfxpl_1
8680 00002E4C 5F                       <1>       pop   edi
8681                                    <1>       ; 08/08/2016
8682                                    <1>       ;cmp  bh, 7
8683                                    <1>       ;jnb  short wgfxpl_3
8684 00002E4D FEC7                     <1>       inc   bh ; i++
8685 00002E4F 3A3D[BE6A0000]           <1>       cmp   bh, [CHAR_HEIGHT]
8686 00002E55 7303                     <1>       jnb   short wgfxpl_3
8687 00002E57 46                       <1>       inc   esi
8688 00002E58 EBC5                     <1>       jmp   short wgfxpl_0
8689                                    <1> wgfxpl_3:
8690                                    <1>       ;mov  dx, 3CEh ; VGAREG_GRDC_ADDRESS
8691 00002E5A 66B808FF                 <1>       mov   ax, 0FF08h
8692 00002E5E 66EF                     <1>       out   dx, ax
8693 00002E60 66B80500                 <1>       mov   ax, 0005h
8694 00002E64 66EF                     <1>       out   dx, ax
8695 00002E66 66B80300                 <1>       mov   ax, 0003h
8696 00002E6A 66EF                     <1>       out   dx, ax
8697                                    <1>       ;
8698 00002E6C 58                       <1>       pop   eax
8699 00002E6D 5A                       <1>       pop   edx
8700 00002E6E 5B                       <1>       pop   ebx
8701 00002E6F 59                       <1>       pop   ecx
8702 00002E70 C3                       <1>       retn
8703                                    <1>
8704                                    <1> vga_write_pixel:
8705                                    <1>       ; 09/07/2016 (TRDOS 386 = TRDOS v2.0)
8706                                    <1>       ;
8707                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
8708                                    <1>       ; vgabios-0.7a (2011)
8709                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
8710                                    <1>       ; 'vgabios.c', 'biosfn_write_pixel'
8711                                    <1>
8712                                    <1>       ; INPUT ->
8713                                    <1>       ;     DX = row (0-239)
8714                                    <1>       ;     CX = column (0-799)
8715                                    <1>       ;     AL = pixel value
8716                                    <1>       ;     (AH = [CRT_MODE])
8717                                    <1>       ; OUTPUT ->
8718                                    <1>       ;     none
8719                                    <1>
8720 00002E71 88C3                     <1>       mov   bl, al ; pixel value
8721                                    <1>       ;mov  ah, [CRT_MODE]
8722 00002E73 BE[D66A0000]             <1>       mov   esi, vga_modes
8723 00002E78 89F7                     <1>       mov   edi, esi
8724 00002E7A 83C710                   <1>       add   edi, vga_mode_count
8725                                    <1> vga_wp_0:
8726 00002E7D AC                       <1>       lodsb
8727 00002E7E 38E0                     <1>       cmp   al, ah ; [CRT_MODE]
8728 00002E80 7405                     <1>       je    short vga_wp_1
8729 00002E82 39FE                     <1>       cmp   esi, edi
8730 00002E84 72F7                     <1>       jb    short vga_wp_0
8731 00002E86 C3                       <1>       retn  ; nothing to do
8732                                    <1> vga_wp_1:
8733 00002E87 83C64F                   <1>       add   esi, vga_memmodel - (vga_modes + 1)
8734                                    <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
8735 00002E8A BF00000A00               <1>       mov   edi, 0A0000h
8736                                    <1>       ;
8737 00002E8F 803E04                   <1>       cmp   byte [esi], PLANAR4
8738 00002E92 741D                     <1>       je    short vga_wp_planar
8739 00002E94 803E03                   <1>       cmp   byte [esi], PLANAR1
8740 00002E97 7418                     <1>       je    short vga_wp_planar
8741                                    <1> vga_wp_linear8:
8742                                    <1>       ; addr=CX+DX*(read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS)*8);
8743 00002E99 0FB605[BC6A0000]         <1>       movzx  eax, byte [CRT_COLS] ; = [VGA_COLS] ; nbcols
8744 00002EA0 66C1E003                 <1>       shl   ax, 3 ; * 8
8745 00002EA4 66F7E2                   <1>       mul   dx
8746 00002EA7 50                       <1>       push  eax
8747                                    <1>       ;mov  edi, 0A0000h
8748 00002EA8 6601CF                   <1>       add   di, cx
8749 00002EAB 58                       <1>       pop   eax
8750 00002EAC 01C7                     <1>       add   edi, eax ; addr
8751                                    <1>       ; write_byte(0xa000,addr,AL);
8752 00002EAE 881F                     <1>       mov   [edi], bl
8753 00002EB0 C3                       <1>       retn
8754                                    <1> vga_wp_planar:
8755                                    <1>       ; addr = CX/8+DX*read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS);
8756 00002EB1 0FB7C1                   <1>       movzx  eax, cx
8757 00002EB4 66C1E803                 <1>       shr   ax, 3 ; CX/8
8758 00002EB8 50                       <1>       push  eax
8759 00002EB9 28E4                     <1>       sub   ah, ah ; 0
8760 00002EBB A0[BC6A0000]             <1>       mov   al, [CRT_COLS] ; = [VGA_COLS] ; nbcols
8761 00002EC0 66F7E2                   <1>       mul   dx
8762                                    <1>       ;mov  edi, 0A0000h
8763 00002EC3 6601C7                   <1>         add     di, ax
8764 00002EC6 58                       <1>       pop   eax
8765 00002EC7 01C7                     <1>       add   edi, eax ; addr
8766 00002EC9 80E107                   <1>       and   cl, 7
8767 00002ECC B580                     <1>       mov   ch, 80h ; mask
8768 00002ECE D2ED                     <1>       shr   ch, cl       ; mask = 0x80 >> (CX & 0x07);
8769                                    <1>
8770                                    <1>       ; outw(VGAREG_GRDC_ADDRESS, (mask << 8) | 0x08);
8771 00002ED0 66BACE03                 <1>       mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
8772 00002ED4 88EC                     <1>       mov   ah, ch
```

```
8773 00002ED6 B008              <1>       mov    al, 8
8774 00002ED8 66EF              <1>       out    dx, ax
8775                            <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0205);
8776 00002EDA 66B80502          <1>       mov    ax, 0205h
8777 00002EDE 66EF              <1>       out    dx, ax
8778                            <1>       ; data = read_byte(0xa000,addr);
8779 00002EE0 8A07              <1>       mov    al, [edi] ; (delay?)
8780                            <1>       ; if (AL & 0x80)
8781                            <1>       ; {
8782                            <1>       ;  outw(VGAREG_GRDC_ADDRESS, 0x1803);
8783                            <1>       ; }
8784 00002EE2 F6C380            <1>       test   bl, 80h
8785 00002EE5 7406              <1>       jz     short vga_wp_2
8786 00002EE7 66B80318          <1>       mov    ax, 1803h
8787 00002EEB 66EF              <1>       out    dx, ax
8788                            <1> vga_wp_2:
8789                            <1>       ; write_byte(0xa000,addr,AL);
8790 00002EED 881F              <1>       mov    [edi], bl
8791                            <1>       ;
8792                            <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
8793 00002EEF 66B808FF          <1>       mov    ax, 0FF08h
8794 00002EF3 66EF              <1>       out    dx, ax
8795 00002EF5 66B80500          <1>       mov    ax, 0005h
8796 00002EF9 66EF              <1>       out    dx, ax
8797 00002EFB 66B80300          <1>       mov    ax, 0003h
8798 00002EFF 66EF              <1>       out    dx, ax
8799                            <1>       ;
8800 00002F01 C3                <1>       retn
8801                            <1>
8802                            <1> vga_read_pixel:
8803                            <1>       ; 09/07/2016 (TRDOS 386 = TRDOS v2.0)
8804                            <1>       ;
8805                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
8806                            <1>       ; vgabios-0.7a (2011)
8807                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
8808                            <1>       ; 'vgabios.c', 'biosfn_read_pixel'
8809                            <1>
8810                            <1>       ; INPUT ->
8811                            <1>       ;      DX = row (0-239)
8812                            <1>       ;      CX = column (0-799)
8813                            <1>       ;      (AH = [CRT_MODE])
8814                            <1>       ; OUTPUT ->
8815                            <1>       ;      AL = pixel value
8816                            <1>
8817                            <1>       ;mov   ah, [CRT_MODE]
8818 00002F02 BE[D66A0000]      <1>       mov    esi, vga_modes
8819 00002F07 89F7              <1>       mov    edi, esi
8820 00002F09 83C710            <1>       add    edi, vga_mode_count
8821                            <1> vga_rp_0:
8822 00002F0C AC                <1>       lodsb
8823 00002F0D 38E0              <1>       cmp    al, ah ; [CRT_MODE]
8824 00002F0F 7405              <1>       je     short vga_rp_1
8825 00002F11 39FE              <1>       cmp    esi, edi
8826 00002F13 72F7              <1>       jb     short vga_rp_0
8827 00002F15 C3                <1>       retn   ; nothing to do
8828                            <1> vga_rp_1:
8829 00002F16 83C64F            <1>       add    esi, vga_memmodel - (vga_modes + 1)
8830                            <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
8831 00002F19 BF00000A00        <1>       mov    edi, 0A0000h
8832                            <1>       ;
8833 00002F1E 803E04            <1>       cmp    byte [esi], PLANAR4
8834 00002F21 741D              <1>       je     short vga_rp_planar
8835 00002F23 803E03            <1>       cmp    byte [esi], PLANAR1
8836 00002F26 7418              <1>       je     short vga_rp_planar
8837                            <1> vga_rp_linear8:
8838                            <1>       ; addr=CX+DX*(read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS)*8);
8839 00002F28 0FB605[BC6A0000]  <1>       movzx  eax, byte [CRT_COLS] ; = [VGA_COLS] ; nbcols
8840 00002F2F 66C1E003          <1>       shl    ax, 3 ; * 8
8841 00002F33 66F7E2            <1>       mul    dx
8842 00002F36 50                <1>       push   eax
8843                            <1>       ;mov   edi, 0A0000h
8844 00002F37 6601CF            <1>       add    di, cx
8845 00002F3A 58                <1>       pop    eax
8846 00002F3B 01C7              <1>       add    edi, eax ; addr
8847                            <1>       ; attr=read_byte(0xa000,addr);
8848 00002F3D 8A07              <1>       mov    al, [edi] ; pixel value
8849 00002F3F C3                <1>       retn
8850                            <1> vga_rp_planar:
8851                            <1>       ; addr = CX/8+DX*read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS);
8852 00002F40 0FB7C1            <1>       movzx  eax, cx
8853 00002F43 66C1E803          <1>       shr    ax, 3 ; CX/8
8854 00002F47 50                <1>       push   eax
8855 00002F48 28E4              <1>       sub    ah, ah ; 0
8856 00002F4A A0[BC6A0000]      <1>       mov    al, [CRT_COLS] ; = [VGA_COLS] ; nbcols
8857 00002F4F 66F7E2            <1>       mul    dx
8858                            <1>       ;mov   edi, 0A0000h
8859 00002F52 6601C7            <1>         add     di, ax
8860 00002F55 58                <1>       pop    eax
8861 00002F56 01C7              <1>       add    edi, eax ; addr
8862 00002F58 80E107            <1>       and    cl, 7
8863 00002F5B B580              <1>       mov    ch, 80h ; mask
8864 00002F5D D2ED              <1>       shr    ch, cl       ; mask = 0x80 >> (CX & 0x07);
8865                            <1>       ; attr = 0x00;
8866 00002F5F 30DB              <1>       xor    bl, bl ; attr = bl = 0,
8867 00002F61 30C9              <1>       xor    cl, cl ; i = cl = 0
8868                            <1>       ; for(i=0;i<4;i++)
8869                            <1>             ; {
8870                            <1>             ;  outw(VGAREG_GRDC_ADDRESS, (i << 8) | 0x04);
8871                            <1>             ;  data = read_byte(0xa000,addr) & mask;
8872                            <1>             ;  if (data > 0) attr |= (0x01 << i);
8873                            <1>             ; }
8874                            <1> vga_rp_2:
8875 00002F63 88CC              <1>       mov    ah, cl ; i << 8
8876 00002F65 B004              <1>       mov    al, 4  ; | 0x04
8877 00002F67 66BACE03          <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
```

```
8878 00002F6B 66EF              <1>       out    dx, ax
8879                            <1>       ; data = read_byte(0xa000,addr) & mask;
8880 00002F6D 8A07              <1>       mov    al, [edi]
8881 00002F6F 20E8              <1>       and    al, ch ; & mask
8882                            <1>       ; if (data > 0) attr |= (0x01 << i);
8883 00002F71 08C0              <1>       or     al, al
8884 00002F73 7408              <1>       jz     short vga_rp_3 ; al = 0
8885 00002F75 B701              <1>       mov    bh, 1
8886 00002F77 D2E7              <1>       shl    bh, cl ; (0x01 << i)
8887 00002F79 08FB              <1>       or     bl, bh ; attr |= (0x01 << i)
8888 00002F7B 88D8              <1>       mov    al, bl ; pixel value
8889                            <1> vga_rp_3:
8890 00002F7D C3                <1>       retn
8891                            <1>
8892                            <1> vga_beeper:
8893                            <1>       ; 04/08/2016  (TRDOS 386 = TRDOS v2.0)
8894 00002F7E FB                <1>       sti
8895                            <1>       ;mov   bh, [ACTIVE_PAGE]
8896 00002F7F E9D1F3FFFF        <1>         jmp     beeper_gfx
8897                            <1>
8898                            <1> vga_write_teletype:
8899                            <1>       ; 12/04/2021 (TRDOS 386 v2.0.3, 32 bit push/pop)
8900                            <1>       ; 09/12/2017
8901                            <1>       ; 06/08/2016
8902                            <1>       ; 04/08/2016
8903                            <1>       ; 01/08/2016
8904                            <1>       ; 31/07/2016
8905                            <1>       ; 09/07/2016 (TRDOS 386 = TRDOS v2.0)
8906                            <1>       ;
8907                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
8908                            <1>       ; vgabios-0.7a (2011)
8909                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
8910                            <1>       ; 'vgabios.c', 'biosfn_write_teletype'
8911                            <1>       ; 'biosfn_write_char_only'
8912                            <1>
8913                            <1>       ; INPUT ->
8914                            <1>       ; [CRT_MODE] = current video mode (>7)
8915                            <1>       ; AL = Character to write
8916                            <1>       ; BL = Color of character
8917                            <1>       ; OUTPUT ->
8918                            <1>       ; Regen buffer updated
8919                            <1>
8920                            <1>       ; biosfn_write_teletype (car, page, attr, flag)
8921                            <1>       ; car = character (AL)
8922                            <1>       ; page = 0
8923                            <1>       ; attr = color (BL)
8924                            <1>       ; 'flag' not used
8925                            <1>
8926 00002F84 8A25[BA6A0000]    <1>       mov    ah, [CRT_MODE]
8927 00002F8A 88C7              <1>       mov    bh, al ; character
8928 00002F8C 668B15[C6800100]  <1>       mov    dx, [CURSOR_POSN] ; cursor pos for page 0
8929                            <1>
8930 00002F93 BE[DE6A0000]      <1>       mov    esi, vga_g_modes
8931 00002F98 89F7              <1>       mov    edi, esi
8932 00002F9A 83C708            <1>       add    edi, vga_g_mode_count
8933                            <1> vga_wtty_0:
8934 00002F9D AC                <1>       lodsb
8935 00002F9E 38E0              <1>       cmp    al, ah ; [CRT_MODE]
8936 00002FA0 7405              <1>       je     short vga_wtty_2
8937 00002FA2 39FE              <1>       cmp    esi, edi
8938 00002FA4 72F7              <1>       jb     short vga_wtty_0
8939                            <1> vga_wtty_1:
8940 00002FA6 C3                <1>       retn   ; nothing to do
8941                            <1> vga_wtty_2:
8942 00002FA7 80FF07            <1>       cmp    bh, 07h ; bell (beep)
8943 00002FAA 74D2              <1>       je     short vga_beeper  ; u11
8944 00002FAC 80FF08            <1>       cmp    bh, 08h ; backspace
8945 00002FAF 7508              <1>       jne    short vga_wtty_3
8946                            <1>       ; if(xcurs>0)xcurs--;
8947 00002FB1 08D2              <1>       or     dl, dl ; xcurs (column)
8948 00002FB3 74F1              <1>       jz     short vga_wtty_1
8949 00002FB5 FECA              <1>       dec    dl ; xcurs--;
8950 00002FB7 EB55              <1>         jmp     short vga_wtty_12
8951                            <1> vga_wtty_3:
8952 00002FB9 80FF0D            <1>       cmp    bh, 0Dh ; carriage return (\r)
8953 00002FBC 7504              <1>       jne    short vga_wtty_4
8954                            <1>       ; xcurs=0;
8955 00002FBE 28D2              <1>       sub    dl, dl ; 0
8956 00002FC0 EB4C              <1>         jmp     short vga_wtty_12
8957                            <1> vga_wtty_4:
8958 00002FC2 80FF0A            <1>       cmp    bh, 0Ah ; new line (\n)
8959 00002FC5 7504              <1>       jne    short vga_wtty_5
8960                            <1>       ; ycurs++;
8961 00002FC7 FEC6              <1>       inc    dh ; next row
8962 00002FC9 EB5E              <1>         jmp     short vga_wtty_11
8963                            <1> vga_wtty_5:
8964 00002FCB 80FF09            <1>       cmp    bh, 09h ; tab stop
8965 00002FCE 7523              <1>       jne    short vga_wtty_8
8966 00002FD0 88D0              <1>       mov    al, dl
8967                            <1>       ;cbw
8968 00002FD2 30E4              <1>       xor    ah, ah ; 09/12/2017
8969 00002FD4 B108              <1>       mov    cl, 8
8970 00002FD6 F6F1              <1>       div    cl
8971 00002FD8 28E1              <1>       sub    cl, ah
8972                            <1>       ;
8973 00002FDA B720              <1>       mov    bh, 20h ; space
8974                            <1> vga_wtty_6: ; tab stop loop
8975                            <1>       ;push  cx
8976                            <1>       ;push  bx
8977                            <1>       ; 12/04/2021
8978 00002FDC 51                <1>       push   ecx
8979 00002FDD 53                <1>       push   ebx
8980 00002FDE E810000000        <1>       call   vga_wtty_8
8981                            <1>       ;pop   bx  ; bh = character, bl = color
8982                            <1>       ;pop   cx
```

```
8983                                    <1>       ; 12/04/2021
8984 00002FE3 5B                        <1>       pop    ebx  ; bh = character, bl = color
8985 00002FE4 59                        <1>       pop    ecx
8986 00002FE5 FEC9                      <1>       dec    cl
8987 00002FE7 7409                      <1>       jz     short vga_wtty_7
8988 00002FE9 668B15[C6800100]          <1>       mov    dx, [CURSOR_POSN] ; new cursor position (pg 0)
8989 00002FF0 EBEA                      <1>       jmp    short vga_wtty_6
8990                                    <1> vga_wtty_7:
8991 00002FF2 C3                        <1>       retn
8992                                    <1>       ;
8993                                    <1> vga_wtty_8:
8994 00002FF3 83C64F                    <1>       add    esi, vga_g_memmodel - (vga_g_modes + 1)
8995                                    <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
8996 00002FF6 BF00000A00                <1>       mov    edi, 0A0000h
8997                                    <1>       ;
8998 00002FFB 88F8                      <1>       mov    al, bh ; character
8999                                    <1>       ;
9000 00002FFD 803E04                    <1>       cmp    byte [esi], PLANAR4
9001 00003000 7414                      <1>       je     short vga_wtty_planar
9002 00003002 803E03                    <1>       cmp    byte [esi], PLANAR1
9003 00003005 740F                      <1>       je     short vga_wtty_planar
9004                                    <1> vga_wtty_linear8:
9005                                    <1>       ; write_gfx_char_lin(car,attr,xcurs,ycurs,nbcols);
9006                                    <1>       ; AL = car, BL = attr (color), DL = xcurs, DH = ycurs,
9007                                    <1>       ; [CRT_COLS] = nbcols
9008 00003007 E8EBFCFFFF                <1>       call   write_gfx_char_lin
9009 0000300C EB0D                      <1>       jmp    short vga_wtty_9
9010                                    <1>
9011                                    <1> vga_wtty_12:
9012                                    <1>       ; 09/07/2016
9013                                    <1>       ; set cursor position
9014                                    <1>       ; NOTE: Hardware cursor position will not be set
9015                                    <1>       ;   in any VGA modes (>7)
9016                                    <1>       ;   But, cursor position will be saved into
9017                                    <1>       ;   [CURSOR_POSN].
9018                                    <1>       ;   TRDOS 386 (TRDOS v2.0) uses only one page
9019                                    <1>       ;   (page 0) for all graphics modes.
9020                                    <1>
9021 0000300E 668915[C6800100]          <1>       mov    [CURSOR_POSN], dx ; save cursor pos for pg 0
9022                                    <1>       ; 04/08/2016
9023                                    <1>       ;mov  bh, [ACTIVE_PAGE] ; = 0
9024                                    <1>       ;call _set_cpos
9025 00003015 C3                        <1>       retn
9026                                    <1>
9027                                    <1> vga_wtty_planar:
9028                                    <1>       ; write_gfx_char_pl4(car,attr,xcurs,ycurs,nbcols,cheight);
9029                                    <1>       ; AL = car, BL = attr (color), DL = xcurs, DH = ycurs,
9030                                    <1>       ; [CRT_COLS]= nbcols, [CHAR_HEIGHT] = cheight
9031 00003016 E87FFDFFFF                <1>       call   write_gfx_char_pl4
9032                                    <1> vga_wtty_9:
9033 0000301B FEC2                      <1>       inc    dl ; xcurs++;
9034                                    <1> vga_wtty_10:
9035                                    <1>       ; Do we need to wrap ?
9036                                    <1>       ; if(xcurs==nbcols)
9037 0000301D 3A15[BC6A0000]            <1>       cmp    dl, [CRT_COLS] ; [VGA_COLS]
9038 00003023 7204                      <1>       jb     short vga_wtty_11 ; no
9039 00003025 28D2                      <1>       sub    dl, dl ; xcurs=0;
9040 00003027 FEC6                      <1>       inc    dh ;  ycurs++;
9041                                    <1> vga_wtty_11:
9042                                    <1>       ; Do we need to scroll ?
9043                                    <1>       ; if(ycurs==nbrows)
9044 00003029 3A35[C26A0000]            <1>       cmp    dh, [VGA_ROWS]
9045 0000302F 72DD                      <1>       jb     short vga_wtty_12 ; no
9046                                    <1>       ;
9047                                    <1>       ; biosfn_scroll (nblines,attr,rul,cul,rlr,clr,page,dir)
9048                                    <1>       ; al = nblines = 1, bl = attr (color) = 0
9049                                    <1>       ; ch = rul, cl = cul, dh = rlr, dl = clr, page = 0
9050                                    <1>       ; dir = SCROLL_UP
9051                                    <1>
9052 00003031 B001                      <1>       mov    al, 1
9053 00003033 28DB                      <1>       sub    bl, bl ; 0 ; blank/black line (attr=0) will be used
9054 00003035 6629C9                    <1>       sub    cx, cx ; 0,0
9055                                    <1>
9056                                    <1>       ; 06/08/2016
9057 00003038 8A35[C26A0000]            <1>       mov    dh, [VGA_ROWS]
9058 0000303E FECE                      <1>       dec    dh ; nbrows -1
9059                                    <1>
9060                                    <1>       ;push  dx ; 04/08/2016
9061                                    <1>       ; 12/04/2021
9062 00003040 52                        <1>       push   edx
9063 00003041 8A15[BC6A0000]            <1>       mov    dl, [CRT_COLS]
9064 00003047 FECA                      <1>       dec    dl ; nbcols -1
9065                                    <1>
9066 00003049 8A25[BA6A0000]            <1>       mov    ah, [CRT_MODE]
9067                                    <1>
9068                                    <1>       ; biosfn_scroll(0x01,0x00,0,0,nbrows-1,nbcols-1,page,SCROLL_UP);
9069 0000304F E810F5FFFF                <1>       call   vga_graphics_up
9070                                    <1>       ; 04/08/2016
9071                                    <1>       ;pop   dx
9072                                    <1>       ; 12/04/2021
9073 00003054 5A                        <1>       pop    edx
9074                                    <1>
9075                                    <1>       ;dec   dh ; ycurs-=1
9076 00003055 EBB7                      <1>       jmp    short vga_wtty_12
9077                                    <1>
9078                                    <1> font_setup:
9079                                    <1>       ; 09/01/2021 (TRDOS 386 v2.0.3)
9080                                    <1>       ; 09/07/2016
9081                                    <1>       ; character generator (font loading) functions
9082                                    <1>       ;
9083                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
9084                                    <1>       ; vgabios-0.7a (2011)
9085                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
9086                                    <1>       ; 'vgabios.c', 'int10_func'
9087                                    <1>
```

```
9088                                    <1>       ; AX = 1100H ; Load User-Defined Font (EGA/VGA)
9089                                    <1>       ;
9090                                    <1>         ; BH    height of each character (bytes per character definition)
9091                                    <1>         ; (BL   font block to load (EGA: 0-3; VGA: 0-7))
9092                                    <1>       ; CX    number of characters to redefine (<=256)
9093                                    <1>         ; DX    ASCII code of the first character defined at ES:BP
9094                                    <1>         ; EBP      address of font-definition information
9095                                    <1>       ;      (in user's memory space)
9096                                    <1>
9097                                    <1>       ; case 0x11:
9098                                    <1>       ; switch(GET_AL())
9099                                    <1>             ; {
9100                                    <1>       ; case 0x00:
9101                                    <1>         ; case 0x10:
9102                                    <1>         ; biosfn_load_text_user_pat(GET_AL(),ES,BP,CX,DX,GET_BL(),GET_BH());
9103                                    <1>         ; break;
9104                                    <1>
9105                                    <1>       ; AX = 1110H ; Load and Activate User-Defined Font (EGA/VGA)
9106 00003057 08C0                     <1>       or     al, al ; 0
9107 00003059 7404                     <1>       jz     short font_setup_0
9108 0000305B 3C10                     <1>       cmp    al, 10h
9109 0000305D 7511                     <1>       jne    short font_setup_1
9110                                    <1> font_setup_0:
9111 0000305F E8CE000000               <1>       call   transfer_user_fonts
9112 00003064 721C                     <1>       jc     short font_setup_error
9113 00003066 E8B2010000               <1>       call   load_text_user_pat
9114 0000306B E979EAFFFF               <1>         jmp     VIDEO_RETURN
9115                                    <1> font_setup_1:
9116                                    <1>       ; AX = 1101H ; Load ROM 8x14 Character Set (EGA/VGA)
9117                                    <1>       ; case 0x01:
9118                                    <1>         ; case 0x11:
9119                                    <1>         ; biosfn_load_text_8_14_pat(GET_AL(),GET_BL());
9120                                    <1>         ; break;
9121 00003070 3C01                     <1>       cmp    al, 1
9122 00003072 7404                     <1>       je     short font_setup_2
9123 00003074 3C11                     <1>       cmp    al, 11h
9124 00003076 7511                     <1>       jne    short font_setup_3
9125                                    <1> font_setup_2:
9126                                    <1>       ; AX = 1111H ; Load and Activate ROM 8x14 Character Set (EGA/VGA)
9127                                    <1>       ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
9128 00003078 E8DC020000               <1>       call   load_text_8_14_pat
9129 0000307D E967EAFFFF               <1>         jmp     VIDEO_RETURN
9130                                    <1> font_setup_error:
9131 00003082 29C0                     <1>       sub    eax, eax ; 0 -> fonts could not be loaded
9132 00003084 E965EAFFFF               <1>       jmp    _video_return
9133                                    <1> font_setup_3:
9134                                    <1>       ; AX = 1102H ; Load ROM 8x8 Character Set (EGA/VGA)
9135                                    <1>       ; case 0x02:
9136                                    <1>         ; case 0x12:
9137                                    <1>         ; biosfn_load_text_8_8_pat(GET_AL(),GET_BL());
9138                                    <1>         ; break;
9139 00003089 3C02                     <1>       cmp    al, 2
9140 0000308B 7404                     <1>       je     short font_setup_4
9141 0000308D 3C12                     <1>       cmp    al, 12h
9142 0000308F 750A                     <1>       jne    short font_setup_5
9143                                    <1> font_setup_4:
9144                                    <1>       ; AX = 1112H ; Load and Activate ROM 8x8 Character Set (EGA/VGA)
9145                                    <1>       ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
9146 00003091 E8F3020000               <1>       call   load_text_8_8_pat
9147 00003096 E94EEAFFFF               <1>         jmp     VIDEO_RETURN
9148                                    <1> font_setup_5:
9149                                    <1>       ; AX = 1104H ; Load ROM 8x16 Character Set (EGA/VGA)
9150                                    <1>       ; case 0x04:
9151                                    <1>         ; case 0x14:
9152                                    <1>         ; biosfn_load_text_8_16_pat(GET_AL(),GET_BL());
9153                                    <1>         ; break;
9154 0000309B 3C04                     <1>       cmp    al, 4
9155 0000309D 7404                     <1>       je     short font_setup_6
9156 0000309F 3C14                     <1>       cmp    al, 14h
9157 000030A1 750A                     <1>       jne    short font_setup_7
9158                                    <1> font_setup_6:
9159                                    <1>       ; AX = 1114H ; Load and Activate ROM 8x16 Character Set (EGA/VGA)
9160                                    <1>       ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
9161 000030A3 E82A030000               <1>       call   load_text_8_16_pat
9162 000030A8 E93CEAFFFF               <1>         jmp     VIDEO_RETURN
9163                                    <1> font_setup_7:
9164                                    <1>       ; Note: AX=1120h (Setup INT 1Fh, EXT_PTR) is not needed
9165                                    <1>       ; for TRDOS 386 (TRDOS v2.0) video functionality;
9166                                    <1>       ; because, originally EXT_PTR (font address) was used for
9167                                    <1>       ; chars 80h to 0FFh (after the first 128 ASCII char fonts), for
9168                                    <1>       ; CGA graphics mode; currenty, 'vgafont8' address has 256 chars!
9169                                    <1>       ;
9170                                    <1>       ; case 0x20:
9171                                    <1>         ; biosfn_load_gfx_8_8_chars(ES,BP);
9172                                    <1>         ; break;
9173                                    <1>       ; case 0x21:
9174                                    <1>         ; biosfn_load_gfx_user_chars(ES,BP,CX,GET_BL(),GET_DL());
9175                                    <1>         ; break;
9176                                    <1>       ; AX = 1121H ; Setup User-Defined Font for Graphics Mode (VGA)
9177                                    <1>       ; BL   screen rows code: 00H = user-specified (in DL)
9178                                    <1>         ;                      01H = 14 rows
9179                                    <1>         ;                      02H = 25 rows
9180                                    <1>         ;                      03H = 43 rows
9181                                    <1>         ; CX   bytes per character definition
9182                                    <1>         ; DL   (when BL=0) custom number of character rows on screen
9183                                    <1>         ; EBP  address of font-definition information (user's mem space)
9184                                    <1>
9185 000030AD 3C21                     <1>       cmp    al, 21h
9186 000030AF 7531                     <1>       jne    short font_setup_9
9187                                    <1>
9188                                    <1>       ; TRDOS 386 modification !
9189                                    <1>       ; dh = 0 -> 256 characters
9190                                    <1>       ; dh = 80h -> second 128 characters
9191                                    <1>       ; dh = 0FFh -> first 128 characters
9192                                    <1>
```

```
9193                                    <1>       ; 09/01/2021 (TRDOS 386 v2.0.3)
9194                                    <1>       ;push  ebx
9195 000030B1 51                        <1>       push   ecx
9196 000030B2 52                        <1>       push   edx
9197 000030B3 30D2                      <1>       xor    dl, dl
9198 000030B5 88CF                      <1>       mov    bh, cl ; character height
9199 000030B7 66B90001                  <1>       mov    cx, 100h ; 256
9200 000030BB 08F6                      <1>       or     dh, dh ; 0
9201 000030BD 7410                      <1>       jz     short  font_setup_8
9202 000030BF FECD                      <1>       dec    ch ; cx = 0
9203 000030C1 80FEFF                    <1>       cmp    dh, 0FFh
9204 000030C4 7409                      <1>       je     short font_setup_8 ; 1st 128 chars
9205                                    <1>       ; 2nd 128 chars
9206 000030C6 80FE80                    <1>       cmp    dh, 80h
9207 000030C9 75B7                      <1>       jne    short font_setup_error ; invalid !
9208 000030CB 88F1                      <1>       mov    cl, dh
9209 000030CD 86D6                      <1>       xchg   dl, dh
9210                                    <1>       ; number of chars, cx = 80h
9211                                    <1>       ; start char, dl = 80h
9212                                    <1> font_setup_8:
9213 000030CF E85E000000                <1>       call   transfer_user_fonts
9214 000030D4 5A                        <1>       pop    edx
9215 000030D5 59                        <1>       pop    ecx
9216                                    <1>       ;pop   ebx
9217 000030D6 72AA                      <1>       jc     short font_setup_error
9218                                    <1>       ; ebp = user's font data address in system's memory space
9219 000030D8 E83F030000                <1>       call   load_gfx_user_chars
9220 000030DD E907EAFFFF                <1>         jmp     VIDEO_RETURN
9221                                    <1> font_setup_9:
9222                                    <1>       ; case 0x22:
9223                                    <1>         ; biosfn_load_gfx_8_14_chars(GET_BL());
9224                                    <1>         ; break;
9225 000030E2 3C22                      <1>       cmp    al, 22h
9226 000030E4 750A                      <1>       jne    short font_setup_10
9227 000030E6 E86D030000                <1>       call   load_gfx_8_14_chars
9228 000030EB E9F9E9FFFF                <1>         jmp     VIDEO_RETURN
9229                                    <1> font_setup_10:
9230                                    <1>       ; case 0x23:
9231                                    <1>         ; biosfn_load_gfx_8_8_dd_chars(GET_BL());
9232                                    <1>         ; break;
9233 000030F0 3C23                      <1>       cmp    al, 23h
9234 000030F2 750A                      <1>       jne    short font_setup_11
9235 000030F4 E8A0030000                <1>       call   load_gfx_8_8_chars
9236 000030F9 E9EBE9FFFF                <1>         jmp     VIDEO_RETURN
9237                                    <1> font_setup_11:
9238                                    <1>       ; case 0x24:
9239                                    <1>         ; biosfn_load_gfx_8_16_chars(GET_BL());
9240                                    <1>         ; break;
9241 000030FE 3C24                      <1>       cmp    al, 24h
9242 00003100 750A                      <1>       jne    short font_setup_12
9243 00003102 E8D3030000                <1>       call   load_gfx_8_16_chars
9244 00003107 E9DDE9FFFF                <1>         jmp     VIDEO_RETURN
9245                                    <1> font_setup_12:
9246                                    <1>       ; case 0x30:
9247                                    <1>         ; biosfn_get_font_info(GET_BH(),&ES,&BP,&CX,&DX);
9248                                    <1>         ; break;
9249 0000310C 3C30                      <1>       cmp    al, 30h
9250 0000310E 750A                      <1>       jne    short font_setup_13
9251 00003110 E806040000                <1>       call   get_font_info
9252                                    <1>       ; eax = return value (info: 4 bytes for 4 parms)
9253                                    <1>       ; eax = 0 -> invalid function (input)
9254 00003115 E9D4E9FFFF                <1>         jmp     _video_return
9255                                    <1> font_setup_13:
9256 0000311A 3C03                      <1>       cmp    al, 03h ; AX = 1103h
9257 0000311C 750D                      <1>       jne    short font_setup_14
9258                                    <1>       ; biosfn_set_text_block_specifier:
9259                                    <1>       ; BL = font block selector code
9260                                    <1>       ; NOTE: TRDOS 386 only uses and sets font block 0
9261                                    <1>       ; (It is as BL = 0 for TRDOS 386)
9262 0000311E 66BAC403                  <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
9263                                    <1>       ;mov   ah, bl
9264 00003122 28E4                      <1>       sub    ah, ah ; 0
9265                                    <1>       ;mov   al, 03h
9266 00003124 66EF                      <1>       out    dx, ax
9267 00003126 E9BEE9FFFF                <1>       jmp     VIDEO_RETURN
9268                                    <1>
9269                                    <1> font_setup_14:
9270 0000312B 29C0                      <1>       sub    eax, eax ; 0 = invalid function
9271 0000312D E9BCE9FFFF                <1>         jmp     _video_return
9272                                    <1>
9273                                    <1> transfer_user_fonts:
9274                                    <1>       ; 19/01/2021
9275                                    <1>       ; 09/01/2021
9276                                    <1>       ; 05/01/2021 (TRDOS 386 v2.0.3)
9277                                    <1>
9278                                    <1>       ; BH    height of each character (bytes per character)
9279                                    <1>       ; CX    number of characters to redefine (<=256)
9280                                    <1>         ; DX    ASCII code of the first character defined at EBP
9281                                    <1>         ; EBP      address of font-definition information
9282                                    <1>       ;      (in user's memory space)
9283                                    <1>
9284                                    <1>       ; Modified registers: eax, edx, ecx, esi, edi, ebp
9285                                    <1>       ;
9286                                    <1>       ; output:
9287                                    <1>       ;      ebp = user font data address in system memory
9288                                    <1>
9289 00003132 81E2FFFF0000              <1>       and    edx, 0FFFFh
9290 00003138 81E1FFFF0000              <1>       and    ecx, 0FFFFh
9291 0000313E 7537                      <1>       jnz    short transfer_user_fonts_5
9292                                    <1>
9293 00003140 09D2                      <1>       or     edx, edx
9294 00003142 7531                      <1>       jnz    short transfer_user_fonts_4
9295 00003144 09ED                      <1>       or     ebp, ebp
9296 00003146 752D                      <1>       jnz    short transfer_user_fonts_4
9297                                    <1>
```

```
                                <1>       ; cx = 0, dx = 0, ebp = 0
                                <1>       ; copy system font to user font
                                <1>
00003148 B140                   <1>       mov    cl, 64 ; 64 dwords
                                <1>
0000314A 80FF10                 <1>       cmp    bh, 16
0000314D 7417                   <1>       je     short transfer_user_fonts_2
0000314F 80FF08                 <1>       cmp    bh, 8
00003152 7406                   <1>       je     short transfer_user_fonts_1
                                <1>
00003154 BD[3C5F0100]           <1>       mov    ebp, vgafont14
00003159 C3                     <1>       retn
                                <1>
                                <1> transfer_user_fonts_1:
0000315A BF00500900             <1>       mov    edi, VGAFONT8USER
0000315F BE[3C570100]           <1>       mov    esi, vgafont8
00003164 EB0A                   <1>       jmp    short transfer_user_fonts_3
                                <1>
                                <1> transfer_user_fonts_2:
00003166 BF00400900             <1>       mov    edi, VGAFONT16USER
0000316B BE[3C6D0100]           <1>       mov    esi, vgafont16
                                <1> transfer_user_fonts_3:
00003170 89FD                   <1>       mov    ebp, edi
00003172 F3A5                   <1>       rep    movsd
00003174 C3                     <1>       retn
                                <1>
                                <1> transfer_user_fonts_4:
00003175 F9                     <1>       stc
00003176 C3                     <1>       retn
                                <1>
                                <1> transfer_user_fonts_5:
00003177 09ED                   <1>       or     ebp, ebp
00003179 74FA                   <1>       jz     short transfer_user_fonts_4 ; invalid address !
                                <1>
0000317B 6681F90001             <1>       cmp    cx, 256
00003180 77F3                   <1>       ja     short transfer_user_fonts_4
00003182 29D1                   <1>       sub    ecx, edx
00003184 76EF                   <1>       jna    short transfer_user_fonts_4
                                <1>
00003186 80FF0E                 <1>       cmp    bh, 14 ; 8x14 font
                                <1>                     ; (there is not an alternative buffer)
00003189 7525                   <1>       jne    short transfer_user_fonts_6
                                <1>
                                <1>       ; use system's 8x14 font space if permission flag is 1
0000318B F605[66120300]80       <1>       test   byte [ufont], 80h
00003192 74E1                   <1>       jz     short transfer_user_fonts_4 ; not allowed
                                <1>
                                <1>       ; permission is given (for vgafont14 location etc.)
                                <1>       ; (for permanent font modification)
                                <1>       ;
                                <1>       ; 19/01/2021
                                <1>       ; Note: Permission flag can be set by 'root' while
                                <1>       ;     system is not in multi tasking/user mode
                                <1>       ;     while [multi_tasking] = 0 and [u.uid] = 0
                                <1>
00003194 52                     <1>       push   edx
00003195 30E4                   <1>       xor    ah, ah
00003197 88F8                   <1>       mov    al, bh ; mov al, 14
00003199 66F7E1                 <1>       mul    cx
                                <1>              ; ecx = byte count
0000319C 89C1                   <1>       mov    ecx, eax
0000319E 5A                     <1>       pop    edx
0000319F 30E4                   <1>       xor    ah, ah
000031A1 88F8                   <1>       mov    al, bh ; mov ax, 14 ; bytes per character
000031A3 66F7E2                 <1>       mul    dx
000031A6 6689C2                 <1>       mov    dx, ax ; char offset
000031A9 BF[3C5F0100]           <1>       mov    edi, vgafont14
000031AE EB4C                   <1>       jmp    short transfer_user_fonts_8
                                <1> transfer_user_fonts_6:
000031B0 80FF08                 <1>       cmp    bh, 8 ; 8x8 font
000031B3 7522                   <1>       jne    short transfer_user_fonts_7 ; 8x16 font
000031B5 66C1E203               <1>       shl    dx, 3 ; * 8
000031B9 66C1E103               <1>       shl    cx, 3 ; * 8
                                <1>       ; 09/01/2021
000031BD BF00500900             <1>       mov    edi, VGAFONT8USER
000031C2 F605[66120300]08       <1>       test   byte [ufont], 8  ; already loaded ?
000031C9 7531                   <1>       jnz    short transfer_user_fonts_8 ; yes
000031CB BE[3C570100]           <1>       mov    esi, vgafont8
000031D0 E83B000000             <1>       call   transfer_user_fonts_10
000031D5 EB25                   <1>       jmp    short transfer_user_fonts_8
                                <1> transfer_user_fonts_7:
000031D7 80FF10                 <1>       cmp    bh, 16 ; 8x16 font
000031DA 7599                   <1>       jne    short transfer_user_fonts_4  ; invalid !
000031DC 66C1E204               <1>       shl    dx, 4 ; * 16
000031E0 66C1E104               <1>       shl    cx, 4 ; * 16
000031E4 BF00400900             <1>       mov    edi, VGAFONT16USER
000031E9 F605[66120300]10       <1>       test   byte [ufont], 16  ; already loaded ?
000031F0 750A                   <1>       jnz    short transfer_user_fonts_8 ; yes
000031F2 BE[3C6D0100]           <1>       mov    esi, vgafont16
000031F7 E814000000             <1>       call   transfer_user_fonts_10
                                <1> transfer_user_fonts_8:
000031FC 01D7                   <1>       add    edi, edx ; char offset
                                <1>       ; 09/07/2016
                                <1>       ;and   ecx, 0FFFFh
                                <1>       ; ECX = byte count
                                <1>       ;push  ecx
000031FE 89EE                   <1>       mov    esi, ebp ; user's font buffer
                                <1>       ; 09/01/2021
00003200 89FD                   <1>       mov    ebp, edi ; system addr for user's font
                                <1>       ; 05/01/2021
                                <1>       ;mov   edi, Cluster_Buffer  ; system buffer
00003202 E885E30000             <1>       call   transfer_from_user_buffer
                                <1>       ;pop   ecx
                                <1>       ; ecx = transfer (byte) count = character count
00003207 7206                   <1>       jc     short transfer_user_fonts_9
```

```
                                <1>       ; 05/01/2021
                                <1>       ;mov   ebp, Cluster_Buffer
                                <1>
00003209 083D[66120300]         <1>       or     byte [ufont], bh
                                <1>                    ; 8x8 or 8x16 user font ready
                                <1> transfer_user_fonts_9:
0000320F C3                     <1>       retn
                                <1>
                                <1> transfer_user_fonts_10:
                                <1>       ; 09/01/2021
00003210 56                     <1>       push   esi
00003211 57                     <1>       push   edi
00003212 51                     <1>       push   ecx
00003213 66B94000               <1>       mov    cx, 64
00003217 F3A5                   <1>       rep    movsd
00003219 59                     <1>       pop    ecx
0000321A 5F                     <1>       pop    edi
0000321B 5E                     <1>       pop    esi
0000321C C3                     <1>       retn
                                <1>
                                <1> load_text_user_pat:
                                <1>       ; 26/07/2016
                                <1>       ; 09/07/2016
                                <1>       ; load user defined (EGA/VGA) text fonts
                                <1>       ;
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
                                <1>       ; 'vgabios.c', 'biosfn_load_text_user_pat'
                                <1>
                                <1>       ; biosfn_load_text_user_pat (AL,ES,BP,CX,DX,BL,BH)
                                <1>
                                <1>       ; get_font_access();
                                <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
                                <1>       ; for(i=0;i<CX;i++)
                                <1>       ; {
                                <1>       ;  src = BP + i * BH;
                                <1>       ;  dest = blockaddr + (DX + i) * 32;
                                <1>       ;  memcpyb(0xA000, dest, ES, src, BH);
                                <1>       ; }
                                <1>       ; release_font_access();
                                <1>       ; if(AL>=0x10)
                                <1>       ; {
                                <1>       ; set_scan_lines(BH);
                                <1>       ; }
                                <1>
0000321D 50                     <1>       push   eax
0000321E E83C000000             <1>       call   get_font_access
00003223 28DB                   <1>       sub    bl, bl ; i = 0
                                <1> ltup_1:
00003225 88D8                   <1>       mov    al, bl
00003227 F6E7                   <1>       mul    bh
00003229 0FB7F0                 <1>       movzx  esi, ax
0000322C 01EE                   <1>       add    esi, ebp
0000322E 88D8                   <1>       mov    al, bl
00003230 28E4                   <1>       sub    ah, ah
00003232 6601D0                 <1>       add    ax, dx ; (DX + i)
00003235 66C1E005               <1>       shl    ax, 5  ; * 32
00003239 0FB7F8                 <1>       movzx  edi, ax
0000323C 81C700000A00           <1>       add    edi, 0A0000h
00003242 51                     <1>       push   ecx
00003243 0FB6CF                 <1>       movzx  ecx, bh
00003246 F3A4                   <1>       rep    movsb
00003248 59                     <1>       pop    ecx
00003249 FEC3                   <1>       inc    bl
0000324B 38CB                   <1>       cmp    bl, cl
0000324D 75D6                   <1>       jne    short ltup_1
                                <1>       ;
0000324F E840000000             <1>       call   release_font_access
                                <1>       ;
00003254 58                     <1>       pop    eax
                                <1>       ; if(AL>=0x10)
00003255 3C10                   <1>       cmp    al, 10h
00003257 7205                   <1>       jb     short ltup_2
                                <1>       ; set_scan_lines(BH);
00003259 E875000000             <1>       call   set_scan_lines
                                <1> ltup_2:
0000325E C3                     <1>       retn
                                <1>
                                <1> get_font_access:
                                <1>       ; 09/07/2016
                                <1>       ;
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
                                <1>       ; 'vgabios.c', 'get_font_access'
                                <1>
                                <1>       ; get_font_access()
0000325F 52                     <1>       push   edx
00003260 66BAC403               <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
00003264 66B80001               <1>       mov    ax, 0100h
00003268 66EF                   <1>       out    dx, ax
0000326A 66B80204               <1>       mov    ax, 0402h
0000326E 66EF                   <1>       out    dx, ax
00003270 66B80407               <1>       mov    ax, 0704h
00003274 66EF                   <1>       out    dx, ax
00003276 66B80003               <1>       mov    ax, 0300h
0000327A 66EF                   <1>       out    dx, ax
0000327C 66BACE03               <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
00003280 66B80402               <1>       mov    ax, 0204h
00003284 66EF                   <1>       out    dx, ax
00003286 66B80500               <1>       mov    ax, 0005h
0000328A 66EF                   <1>       out    dx, ax
0000328C 66B80604               <1>       mov    ax, 0406h
00003290 66EF                   <1>       out    dx, ax
```

```
9508 00003292 5A                <1>       pop    edx
9509 00003293 C3                <1>       retn
9510                            <1>
9511                            <1> release_font_access:
9512                            <1>       ; 29/07/2016
9513                            <1>       ; 09/07/2016
9514                            <1>       ;
9515                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
9516                            <1>       ; vgabios-0.7a (2011)
9517                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
9518                            <1>       ; 'vgabios.c', 'release_font_access'
9519                            <1>
9520 00003294 66BAC403          <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
9521 00003298 66B80001          <1>       mov    ax, 0100h
9522 0000329C 66EF              <1>       out    dx, ax
9523 0000329E 66B80203          <1>       mov    ax, 0302h
9524 000032A2 66EF              <1>       out    dx, ax
9525 000032A4 66B80403          <1>       mov    ax, 0304h
9526 000032A8 66EF              <1>       out    dx, ax
9527 000032AA 66B80003          <1>       mov    ax, 0300h
9528 000032AE 66EF              <1>       out    dx, ax
9529 000032B0 66BACC03          <1>       mov    dx, 3CCh ; VGAREG_READ_MISC_OUTPUT
9530 000032B4 EC                <1>       in     al, dx
9531 000032B5 2401              <1>       and    al, 01h
9532 000032B7 C0E002            <1>       shl    al, 2
9533 000032BA 0C0A              <1>       or     al, 0Ah
9534 000032BC 88C4              <1>       mov    ah, al
9535 000032BE B006              <1>       mov    al, 06h
9536 000032C0 66BACE03          <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
9537 000032C4 66EF              <1>       out    dx, ax
9538 000032C6 66B80400          <1>       mov    ax, 0004h
9539 000032CA 66EF              <1>       out    dx, ax
9540 000032CC 66B80510          <1>       mov    ax, 1005h
9541 000032D0 66EF              <1>       out    dx, ax
9542 000032D2 C3                <1>       retn
9543                            <1>
9544                            <1> set_scan_lines:
9545                            <1>       ; 09/07/2016
9546                            <1>       ;
9547                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
9548                            <1>       ; vgabios-0.7a (2011)
9549                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
9550                            <1>       ; 'vgabios.c', 'set_scan_lines'
9551                            <1>
9552                            <1>       ; set_scan_lines(lines)
9553                            <1>       ; BH = lines
9554                            <1>
9555                            <1>       ; outb(crtc_addr, 0x09);
9556 000032D3 66BAD403          <1>       mov    dx, 3D4h ; CRTC_ADDRESS = 3D4h (always)
9557 000032D7 B009              <1>       mov    al, 09h
9558 000032D9 EE                <1>       out    dx, al
9559                            <1>       ; crtc_r9 = inb(crtc_addr+1);
9560 000032DA 6642              <1>       inc    dx ; 3D5h
9561 000032DC EC                <1>       in     al, dx
9562                            <1>       ; crtc_r9 = (crtc_r9 & 0xe0) | (lines - 1);
9563 000032DD 24E0              <1>       and    al, 0E0h
9564 000032DF FECF              <1>       dec    bh ; lines - 1
9565 000032E1 08F8              <1>       or     al, bh
9566                            <1>       ; outb(crtc_addr+1, crtc_r9);
9567 000032E3 EE                <1>       out    dx, al
9568                            <1>       ;inc  bh
9569                            <1>       ; if(lines==8)
9570                            <1>       ;cmp  bh, 8
9571 000032E4 80FF07            <1>       cmp    bh, 7
9572 000032E7 7506              <1>       jne    short ssl_1
9573                            <1>       ; biosfn_set_cursor_shape(0x06,0x07);
9574 000032E9 66B90706          <1>       mov    cx, 0607h
9575 000032ED EB06              <1>       jmp    short ssl_2
9576                            <1> ssl_1:
9577                            <1>       ; biosfn_set_cursor_shape(lines-4,lines-3);
9578 000032EF 88F9              <1>       mov    cl, bh ; lines - 1
9579 000032F1 88CD              <1>       mov    ch, cl ; lines - 1 (16 -> 15)
9580 000032F3 FECD              <1>       dec    ch  ; lines - 2 (16 -> 14)
9581                            <1> ssl_2:
9582                            <1>       ; CH = start line, CL = stop line
9583 000032F5 B40A              <1>       mov    ah, 10 ; 6845 register for cursor set
9584 000032F7 66890D[D36A0000]  <1>       mov    [CURSOR_MODE], cx ; save in data area
9585 000032FE E822F0FFFF        <1>       call   m16    ; output cx register
9586                            <1>       ; write_word(BIOSMEM_SEG,BIOSMEM_CHAR_HEIGHT, lines);
9587 00003303 FEC7              <1>       inc    bh ; lines
9588 00003305 883D[BE6A0000]    <1>       mov    [CHAR_HEIGHT], bh
9589                            <1>       ;  outb(crtc_addr, 0x12);
9590 0000330B 66BAD403          <1>       mov    dx, 3D4h ; CRTC_ADDRESS
9591 0000330F B012              <1>       mov    al, 12h
9592 00003311 EE                <1>       out    dx, al
9593                            <1>       ; vde = inb(crtc_addr+1);
9594 00003312 6642              <1>       inc    dx
9595 00003314 EC                <1>       in     al, dx
9596 00003315 88C4              <1>       mov    ah, al
9597                            <1>       ; outb(crtc_addr, 0x07);
9598 00003317 664A              <1>       dec    dx
9599 00003319 B007              <1>       mov    al, 07h
9600 0000331B EE                <1>       out    dx, al
9601                            <1>       ; ovl = inb(crtc_addr+1);
9602 0000331C 6642              <1>       inc    dx
9603 0000331E EC                <1>       in     al, dx
9604                            <1>       ; vde += (((ovl & 0x02) << 7) + ((ovl & 0x40) << 3) + 1);
9605 0000331F 88E2              <1>       mov    dl, ah ; vde
9606 00003321 88C6              <1>       mov    dh, al ; ovl
9607 00003323 6683E002          <1>       and    ax, 02h
9608 00003327 66C1E007          <1>       shl    ax, 7
9609 0000332B 6689C1            <1>       mov    cx, ax ; (ovl & 0x02) << 7)
9610 0000332E 88F0              <1>       mov    al, dh ; ovl
9611 00003330 6683E040          <1>       and    ax, 40h
9612 00003334 66C1E003          <1>       shl    ax, 3  ; (ovl & 0x40) << 3)
```

```
9613 00003338 6640              <1>       inc    ax ; + 1
9614 0000333A 6601C8            <1>       add    ax, cx
9615 0000333D 30F6              <1>       xor    dh, dh
9616 0000333F 6601D0            <1>       add    ax, dx ; + vde
9617                            <1>       ; rows = vde / lines;
9618 00003342 F6F7              <1>       div    bh
9619                            <1>       ;dec   al ; rows -1
9620                            <1>       ; write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, rows-1);
9621 00003344 A2[C26A0000]      <1>       mov    [VGA_ROWS], al ; rows (not 'rows-1' !)
9622                            <1>       ; write_word(BIOSMEM_SEG,BIOSMEM_PAGE_SIZE, rows * cols * 2);
9623                            <1>       ;mov   ah, [CRT_COLS]
9624                            <1>       ;mul   ah
9625                            <1>       ; 17/11/2020
9626 00003349 F625[BC6A0000]    <1>       mul    byte [CRT_COLS]
9627 0000334F 66D1E0            <1>       shl    ax, 1
9628 00003352 66A3[408D0100]    <1>       mov    [CRT_LEN], ax
9629 00003358 C3                <1>       retn
9630                            <1>
9631                            <1> load_text_8_14_pat:
9632                            <1>       ; 26/07/2016
9633                            <1>       ; 25/07/2016
9634                            <1>       ; 23/07/2016
9635                            <1>       ; 09/07/2016
9636                            <1>       ; load user defined (EGA/VGA) text fonts
9637                            <1>       ;
9638                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
9639                            <1>       ; vgabios-0.7a (2011)
9640                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
9641                            <1>       ; 'vgabios.c', 'biosfn_load_text_8_14_pat'
9642                            <1>
9643                            <1>       ; biosfn_load_text_8_14_pat (AL,BL)
9644                            <1>
9645                            <1>       ; get_font_access();
9646                            <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
9647                            <1>       ; for(i=0;i<0x100;i++)
9648                            <1>       ; {
9649                            <1>       ;  src = i * 14;
9650                            <1>       ;  dest = blockaddr + i * 32;
9651                            <1>       ;  memcpyb(0xA000, dest, 0xC000, vgafont14+src, 14);
9652                            <1>       ; }
9653                            <1>       ; release_font_access();
9654                            <1>       ; if(AL>=0x10)
9655                            <1>       ; {
9656                            <1>       ; set_scan_lines(14);
9657                            <1>       ; }
9658                            <1>
9659 00003359 50                 <1>       push   eax
9660 0000335A E800FFFFFF       <1>       call   get_font_access
9661                            <1>
9662                            <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
9663                            <1>       ;mov   dl, bl
9664                            <1>       ;and   dl, 3
9665                            <1>       ;shl   dx, 14
9666                            <1>       ;xchg  dx, bx
9667                            <1>       ;and   dl, 4
9668                            <1>       ;shl   dx, 11
9669                            <1>       ;add   dx, bx
9670                            <1>
9671                            <1>       ;xor   dx, dx  ; blockaddr = 0
9672                            <1>       ; Always block 0 for TRDOS 386 ! (blockaddr=0)
9673                            <1>
9674 0000335F 28DB             <1>       sub    bl, bl ; i = 0
9675 00003361 B70E             <1>       mov    bh, 14
9676 00003363 BE[3C5F0100]     <1>       mov    esi, vgafont14
9677 00003368 BF00000A00       <1>       mov    edi, 0A0000h
9678                            <1> lt8_14_1:
9679                            <1>       ;mov   al, bl
9680                            <1>       ;mul   bh
9681                            <1>       ;movzx esi, ax
9682                            <1>       ;add   esi, vgafont14
9683                            <1>       ;mov   al, bl
9684                            <1>       ;sub   ah, ah
9685                            <1>       ;shl   ax, 5 ; * 32
9686                            <1>       ;;add  ax, dx ; blockaddr + i * 32;
9687                            <1>       ;movzx edi, ax ; dest
9688                            <1>       ;add   edi, 0A0000h
9689 0000336D 0FB6CF           <1>       movzx  ecx, bh
9690 00003370 F3A4             <1>       rep    movsb
9691 00003372 83C712           <1>       add    edi, 18 ; 32 - 14
9692 00003375 FEC3             <1>       inc    bl
9693 00003377 75F4             <1>       jnz    short lt8_14_1
9694                            <1>       ;
9695 00003379 E816FFFFFF       <1>       call   release_font_access
9696                            <1>       ;
9697 0000337E 58               <1>       pop    eax
9698                            <1>       ; if(AL>=0x10)
9699 0000337F 3C10             <1>       cmp    al, 10h
9700 00003381 7205             <1>       jb     short lt8_14_4
9701                            <1>       ; BH = 14
9702                            <1>       ; set_scan_lines(14);
9703 00003383 E84BFFFFFF       <1>       call   set_scan_lines
9704                            <1> lt8_14_4:
9705 00003388 C3               <1>       retn
9706                            <1>
9707                            <1> load_text_8_8_pat:
9708                            <1>       ; 05/01/2021 (TRDOS 386 v2.0.3)
9709                            <1>       ; 26/07/2016
9710                            <1>       ; 25/07/2016
9711                            <1>       ; 23/07/2016
9712                            <1>       ; 09/07/2016
9713                            <1>       ; load user defined (EGA/VGA) text fonts
9714                            <1>       ;
9715                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
9716                            <1>       ; vgabios-0.7a (2011)
9717                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
```

```
                                <1>       ; 'vgabios.c', 'biosfn_load_text_8_8_pat'
                                <1>
                                <1>       ; biosfn_load_text_8_8_pat (AL,BL)
                                <1>
                                <1>       ; get_font_access();
                                <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
                                <1>       ; for(i=0;i<0x100;i++)
                                <1>       ; {
                                <1>       ;  src = i * 8;
                                <1>       ;  dest = blockaddr + i * 32;
                                <1>       ;  memcpyb(0xA000, dest, 0xC000, vgafont8+src, 8);
                                <1>       ; }
                                <1>       ; release_font_access();
                                <1>       ; if(AL>=0x10)
                                <1>       ; {
                                <1>       ; set_scan_lines(8);
                                <1>       ; }
                                <1>
00003389 50                     <1>       push   eax
0000338A E8D0FEFFFF             <1>       call   get_font_access
                                <1>
                                <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
                                <1>       ;mov   dl, bl
                                <1>       ;and   dl, 3
                                <1>       ;shl   dx, 14
                                <1>       ;xchg  dx, bx
                                <1>       ;and   dl, 4
                                <1>       ;shl   dx, 11
                                <1>       ;add   dx, bx
                                <1>
                                <1>       ;xor   dx, dx  ; blockaddr = 0
                                <1>       ; Always block 0 for TRDOS 386 ! (blockaddr=0)
                                <1>
0000338F 28DB                   <1>       sub    bl, bl ; i = 0
00003391 B708                   <1>       mov    bh, 8
                                <1>       ;mov   esi, vgafont8
00003393 BF00000A00             <1>       mov    edi, 0A0000h
                                <1>
                                <1>       ; 05/01/2021
00003398 F605[66120300]80       <1>       test   byte [ufont], 80h
0000339F 7410                   <1>       jz     short lt8_8_0
                                <1>              ; user font permission (after set mode)
000033A1 F605[66120300]08       <1>       test   byte [ufont], 8
000033A8 7407                   <1>       jz     short lt8_8_0
000033AA BE00500900             <1>       mov    esi, VGAFONT8USER
000033AF EB05                   <1>       jmp    short lt8_8_1
                                <1> lt8_8_0:
000033B1 BE[3C570100]           <1>       mov    esi, vgafont8
                                <1> lt8_8_1:
                                <1>       ;mov   al, bl
                                <1>       ;mul   bh
                                <1>       ;movzx esi, ax
                                <1>       ;add   esi, vgafont8
                                <1>       ;mov   al, bl
                                <1>       ;sub   ah, ah
                                <1>       ;shl   ax, 5 ; * 32
                                <1>       ;;add  ax, dx ; blockaddr + i * 32;
                                <1>       ;movzx edi, ax ; dest
                                <1>       ;add   edi, 0A0000h
000033B6 0FB6CF                 <1>       movzx  ecx, bh
000033B9 F3A4                   <1>       rep    movsb
000033BB 83C718                 <1>       add    edi, 24 ; 32 - 8
000033BE FEC3                   <1>       inc    bl
000033C0 75F4                   <1>       jnz    short lt8_8_1
                                <1>       ;
000033C2 E8CDFEFFFF             <1>       call   release_font_access
                                <1>       ;
000033C7 58                     <1>       pop    eax
                                <1>       ; if(AL>=0x10)
000033C8 3C10                   <1>       cmp    al, 10h
000033CA 7205                   <1>       jb     short lt8_8_2
                                <1>       ; BH = 8
                                <1>       ; set_scan_lines(8);
000033CC E802FFFFFF             <1>       call   set_scan_lines
                                <1> lt8_8_2:
000033D1 C3                     <1>       retn
                                <1>
                                <1> load_text_8_16_pat:
                                <1>       ; 05/01/2021 (TRDOS 386 v2.0.3)
                                <1>       ; 26/07/2016
                                <1>       ; 25/07/2016
                                <1>       ; 23/07/2016
                                <1>       ; 09/07/2016
                                <1>       ; load user defined (EGA/VGA) text fonts
                                <1>       ;
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
                                <1>       ; 'vgabios.c', 'biosfn_load_text_8_16_pat'
                                <1>
                                <1>       ; biosfn_load_text_8_16_pat (AL,BL)
                                <1>
                                <1>       ; get_font_access();
                                <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
                                <1>       ; for(i=0;i<0x100;i++)
                                <1>       ; {
                                <1>       ;  src = i * 16;
                                <1>       ;  dest = blockaddr + i * 32;
                                <1>       ;  memcpyb(0xA000, dest, 0xC000, vgafont16+src, 16);
                                <1>       ; }
                                <1>       ; release_font_access();
                                <1>       ; if(AL>=0x10)
                                <1>       ; {
                                <1>       ; set_scan_lines(16);
                                <1>       ; }
```

```
9823                                    <1>
9824 000033D2 50                        <1>         push   eax
9825 000033D3 E887FEFFFF                <1>         call   get_font_access
9826                                    <1>
9827                                    <1>         ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
9828                                    <1>         ;mov   dl, bl
9829                                    <1>         ;and   dl, 3
9830                                    <1>         ;shl   dx, 14
9831                                    <1>         ;xchg  dx, bx
9832                                    <1>         ;and   dl, 4
9833                                    <1>         ;shl   dx, 11
9834                                    <1>         ;add   dx, bx
9835                                    <1>
9836                                    <1>         ;xor   dx, dx  ; blockaddr = 0
9837                                    <1>         ; Always block 0 for TRDOS 386 ! (blockaddr=0)
9838                                    <1>
9839 000033D8 28DB                     <1>         sub    bl, bl ; i = 0
9840 000033DA B710                     <1>         mov    bh, 16
9841                                    <1>         ;mov   esi, vgafont16
9842 000033DC BF00000A00               <1>         mov    edi, 0A0000h
9843 000033E1 0FB6C7                   <1>         movzx  eax, bh
9844                                    <1>
9845                                    <1>         ; 05/01/2021
9846 000033E4 F605[66120300]80         <1>         test   byte [ufont], 80h
9847 000033EB 7410                     <1>         jz     short lt8_16_0
9848                                    <1>                ; user font permission (after set mode)
9849 000033ED F605[66120300]10         <1>         test   byte [ufont], 16
9850 000033F4 7407                     <1>         jz     short lt8_16_0
9851 000033F6 BE00400900               <1>         mov    esi, VGAFONT16USER
9852 000033FB EB05                     <1>         jmp    short lt8_16_1
9853                                    <1> lt8_16_0:
9854 000033FD BE[3C6D0100]             <1>         mov    esi, vgafont16
9855                                    <1> lt8_16_1:
9856                                    <1>         ;mov   al, bl
9857                                    <1>         ;mul   bh
9858                                    <1>         ;movzx esi, ax
9859                                    <1>         ;add   esi, vgafont16
9860                                    <1>         ;mov   al, bl ; i
9861                                    <1>         ;sub   ah, ah
9862                                    <1>         ;shl   ax, 5 ; * 32
9863                                    <1>         ;;add  ax, dx ; blockaddr + i * 32;
9864                                    <1>         ;movzx edi, ax ; dest
9865                                    <1>         ;add   edi, 0A0000h
9866                                    <1>         ;movzx ecx, bh
9867 00003402 89C1                     <1>         mov    ecx, eax ; 16
9868 00003404 F3A4                     <1>         rep    movsb
9869 00003406 01C7                     <1>         add    edi, eax ; add edi, 16
9870 00003408 FEC3                     <1>         inc    bl
9871 0000340A 75F6                     <1>         jnz    short lt8_16_1
9872                                    <1>         ;
9873 0000340C E883FEFFFF               <1>         call   release_font_access
9874                                    <1>         ;
9875 00003411 58                       <1>         pop    eax
9876                                    <1>         ; if(AL>=0x10)
9877 00003412 3C10                     <1>         cmp    al, 10h
9878 00003414 7205                     <1>         jb     short lt8_16_2
9879                                    <1>         ; BH = 16
9880                                    <1>         ; set_scan_lines(16);
9881 00003416 E8B8FEFFFF               <1>         call   set_scan_lines
9882                                    <1> lt8_16_2:
9883 0000341B C3                       <1>         retn
9884                                    <1>
9885                                    <1> load_gfx_user_chars:
9886                                    <1>         ; 08/01/2021
9887                                    <1>         ; 05/01/2021 (TRDOS 386 v2.0.3)
9888                                    <1>         ; 08/08/2016
9889                                    <1>         ; 10/07/2016
9890                                    <1>         ; Setup User-Defined Font for Graphics Mode (VGA)
9891                                    <1>         ;
9892                                    <1>         ; derived from 'Plex86/Bochs VGABios' source code
9893                                    <1>         ; vgabios-0.7a (2011)
9894                                    <1>         ; by the LGPL VGABios developers Team (2001-2008)
9895                                    <1>         ; 'vgabios.c', 'biosfn_load_gfx_user_chars'
9896                                    <1>
9897                                    <1>         ; biosfn_load_gfx_user_chars (ES,BP,CX,BL,DL)
9898                                    <1>         ; /* set 0x43 INT pointer */
9899                                    <1>         ; write_word(0x0, 0x43*4, BP);
9900                                    <1>         ; write_word(0x0, 0x43*4+2, ES);
9901                                    <1>
9902                                    <1>         ; 08/01/2021
9903                                    <1>
9904                                    <1>         ; BL   screen rows code: 00H = user-specified (in DL)
9905                                    <1>           ;                      01H = 14 rows
9906                                    <1>           ;                      02H = 25 rows
9907                                    <1>           ;                      03H = 43 rows
9908                                    <1>           ; CX   bytes per character definition
9909                                    <1>           ; DL   (when BL=0) custom number of character rows on screen
9910                                    <1>           ; EBP  address of font-definition information (user's mem space)
9911                                    <1>
9912                                    <1>         ; 05/01/2021
9913                                    <1>         ;xor   eax, eax
9914                                    <1>         ;dec   eax ; 0FFFFFFFFh (user defined fonts)
9915                                    <1>         ;mov   [VGA_INT43H], eax
9916                                    <1>
9917                                    <1>         ; 08/01/2021
9918                                    <1>         ; ebp = video font data (buffer) address
9919 0000341C 892D[528D0100]           <1>         mov    [VGA_INT43H], ebp
9920                                    <1>
9921                                    <1>         ; switch (BL) {
9922                                    <1>         ; case 0:
9923                                    <1>         ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, DL-1);
9924                                    <1>         ;    break;
9925 00003422 20DB                     <1>         and    bl, bl
9926 00003424 7508                     <1>         jnz    short l_gfx_uc_1
9927 00003426 8815[C26A0000]           <1>         mov    [VGA_ROWS], dl  ; not DL-1 !
```

```
9928 0000342C EB23                <1>       jmp    short l_gfx_uc_4
9929                              <1> l_gfx_uc_1:
9930                              <1>       ; case 1:
9931                              <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
9932                              <1>       ;    break;
9933 0000342E FECB                <1>       dec    bl
9934 00003430 7509                <1>       jnz    short l_gfx_uc_2
9935                              <1>       ; bl = 1
9936 00003432 C605[C26A0000]0E    <1>       mov    byte [VGA_ROWS], 14  ; not 13 !
9937 00003439 EB16                <1>       jmp    short l_gfx_uc_4
9938                              <1> l_gfx_uc_2:
9939 0000343B FECB                <1>       dec    bl
9940 0000343D 740B                <1>       jz     short l_gfx_uc_3 ; bl = 2
9941 0000343F FECB                <1>       dec    bl
9942 00003441 750E                <1>       jnz    short l_gfx_uc_4 ; bl > 3
9943                              <1>       ; bl = 3
9944                              <1>       ; case 3:
9945                              <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
9946                              <1>       ;    break;
9947 00003443 C605[C26A0000]2B    <1>       mov    byte [VGA_ROWS], 43  ; not 42 !
9948                              <1> l_gfx_uc_3:
9949                              <1>       ; case 2:
9950                              <1>       ; default:
9951                              <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
9952                              <1>       ;    break;
9953                              <1>       ; bl = 2 or bl > 3
9954 0000344A C605[C26A0000]19    <1>       mov    byte [VGA_ROWS], 25  ; not 24 !
9955                              <1>       ; }
9956                              <1> l_gfx_uc_4:
9957                              <1>       ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, CX);
9958 00003451 880D[BE6A0000]      <1>       mov    [CHAR_HEIGHT], cl
9959                              <1>       ; }
9960 00003457 C3                  <1>       retn
9961                              <1>
9962                              <1> load_gfx_8_14_chars:
9963                              <1>       ; 08/08/2016
9964                              <1>       ; 10/07/2016
9965                              <1>       ; Setup ROM 8x14 Font for Graphics Mode (VGA)
9966                              <1>       ;
9967                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
9968                              <1>       ; vgabios-0.7a (2011)
9969                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
9970                              <1>       ; 'vgabios.c', 'biosfn_load_gfx_8_14_chars'
9971                              <1>
9972                              <1>       ; biosfn_load_gfx_8_14_chars (BL)
9973                              <1>       ; /* set 0x43 INT pointer */
9974                              <1>       ; write_word(0x0, 0x43*4, &vgafont14);
9975                              <1>       ; write_word(0x0, 0x43*4+2, 0xC000);
9976 00003458 C705[528D0100]-     <1>       mov    dword [VGA_INT43H], vgafont14
9976 0000345E [3C5F0100]          <1>
9977                              <1>
9978                              <1>       ; BL    screen rows code: 00H = user-specified (in DL)
9979                              <1>         ;                       01H = 14 rows
9980                              <1>         ;                       02H = 25 rows
9981                              <1>         ;                       03H = 43 rows
9982                              <1>         ; DL    (when BL=0) custom number of char rows on screen
9983                              <1>
9984                              <1>       ; switch (BL) {
9985                              <1>       ; case 0:
9986                              <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, DL-1);
9987                              <1>       ;    break;
9988 00003462 20DB                <1>       and    bl, bl
9989 00003464 7508                <1>       jnz    short l_gfx_8_14c_1
9990 00003466 8815[C26A0000]      <1>       mov    [VGA_ROWS], dl  ; not DL-1 !
9991 0000346C EB23                <1>       jmp    short l_gfx_8_14c_4
9992                              <1> l_gfx_8_14c_1:
9993                              <1>       ; case 1:
9994                              <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
9995                              <1>       ;    break;
9996 0000346E FECB                <1>       dec    bl
9997 00003470 7509                <1>       jnz    short l_gfx_8_14c_2
9998                              <1>       ; bl = 1
 9999 00003472 C605[C26A0000]0E    <1>       mov    byte [VGA_ROWS], 14  ; not 13 !
10000 00003479 EB16                <1>       jmp    short l_gfx_8_14c_4
10001                              <1> l_gfx_8_14c_2:
10002 0000347B FECB                <1>       dec    bl
10003 0000347D 740B                <1>       jz     short l_gfx_8_14c_3 ; bl = 2
10004 0000347F FECB                <1>       dec    bl
10005 00003481 750E                <1>       jnz    short l_gfx_8_14c_4 ; bl > 3
10006                              <1>       ; bl = 3
10007                              <1>       ; case 3:
10008                              <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
10009                              <1>       ;    break;
10010 00003483 C605[C26A0000]2B    <1>       mov    byte [VGA_ROWS], 43  ; not 42 !
10011                              <1> l_gfx_8_14c_3:
10012                              <1>       ; case 2:
10013                              <1>       ; default:
10014                              <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
10015                              <1>       ;    break;
10016                              <1>       ; bl = 2 or bl > 3
10017 0000348A C605[C26A0000]19    <1>       mov    byte [VGA_ROWS], 25  ; not 24 !
10018                              <1>       ; }
10019                              <1> l_gfx_8_14c_4:
10020                              <1>       ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, 14);
10021 00003491 C605[BE6A0000]0E    <1>         mov     byte [CHAR_HEIGHT], 14
10022                              <1>       ; }
10023 00003498 C3                  <1>       retn
10024                              <1>
10025                              <1> load_gfx_8_8_chars:
10026                              <1>       ; 08/08/2016
10027                              <1>       ; 10/07/2016
10028                              <1>       ; Setup ROM 8x14 Font for Graphics Mode (VGA)
10029                              <1>       ;
10030                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
10031                              <1>       ; vgabios-0.7a (2011)
```

```
10032                                 <1>       ; by the LGPL VGABios developers Team (2001-2008)
10033                                 <1>       ; 'vgabios.c', 'biosfn_load_gfx_8_8_dd_chars'
10034                                 <1>
10035                                 <1>       ; biosfn_load_gfx_8_8_dd_chars (BL)
10036                                 <1>       ; /* set 0x43 INT pointer */
10037                                 <1>       ; write_word(0x0, 0x43*4, &vgafont8);
10038                                 <1>       ; write_word(0x0, 0x43*4+2, 0xC000);
10039 00003499 C705[528D0100]-        <1>       mov    dword [VGA_INT43H], vgafont8
10039 0000349F [3C570100]             <1>
10040                                 <1>
10041                                 <1>       ; BL    screen rows code: 00H = user-specified (in DL)
10042                                 <1>         ;                       01H = 14 rows
10043                                 <1>         ;                       02H = 25 rows
10044                                 <1>         ;                       03H = 43 rows
10045                                 <1>         ; DL    (when BL=0) custom number of char rows on screen
10046                                 <1>
10047                                 <1>       ; switch (BL) {
10048                                 <1>       ; case 0:
10049                                 <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, DL-1);
10050                                 <1>       ;    break;
10051 000034A3 20DB                  <1>       and    bl, bl
10052 000034A5 7508                  <1>       jnz    short l_gfx_8_8c_1
10053 000034A7 8815[C26A0000]        <1>       mov    [VGA_ROWS], dl  ; not DL-1 !
10054 000034AD EB23                  <1>       jmp    short l_gfx_8_8c_4
10055                                 <1> l_gfx_8_8c_1:
10056                                 <1>       ; case 1:
10057                                 <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
10058                                 <1>       ;    break;
10059 000034AF FECB                  <1>       dec    bl
10060 000034B1 7509                  <1>       jnz    short l_gfx_8_8c_2
10061                                 <1>       ; bl = 1
10062 000034B3 C605[C26A0000]0E      <1>       mov    byte [VGA_ROWS], 14  ; not 13 !
10063 000034BA EB16                  <1>       jmp    short l_gfx_8_8c_4
10064                                 <1> l_gfx_8_8c_2:
10065 000034BC FECB                  <1>       dec    bl
10066 000034BE 740B                  <1>       jz     short l_gfx_8_8c_3 ; bl = 2
10067 000034C0 FECB                  <1>       dec    bl
10068 000034C2 750E                  <1>       jnz    short l_gfx_8_8c_4 ; bl > 3
10069                                 <1>       ; bl = 3
10070                                 <1>       ; case 3:
10071                                 <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
10072                                 <1>       ;    break;
10073 000034C4 C605[C26A0000]2B      <1>       mov    byte [VGA_ROWS], 43  ; not 42 !
10074                                 <1> l_gfx_8_8c_3:
10075                                 <1>       ; case 2:
10076                                 <1>       ; default:
10077                                 <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
10078                                 <1>       ;    break;
10079                                 <1>       ; bl = 2 or bl > 3
10080 000034CB C605[C26A0000]19      <1>       mov    byte [VGA_ROWS], 25  ; not 24 !
10081                                 <1>       ; }
10082                                 <1> l_gfx_8_8c_4:
10083                                 <1>       ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, 8);
10084 000034D2 C605[BE6A0000]08      <1>         mov     byte [CHAR_HEIGHT], 8
10085                                 <1>       ; }
10086 000034D9 C3                    <1>       retn
10087                                 <1>
10088                                 <1> load_gfx_8_16_chars:
10089                                 <1>       ; 08/08/2016
10090                                 <1>       ; 10/07/2016
10091                                 <1>       ; Setup ROM 8x14 Font for Graphics Mode (VGA)
10092                                 <1>       ;
10093                                 <1>       ; derived from 'Plex86/Bochs VGABios' source code
10094                                 <1>       ; vgabios-0.7a (2011)
10095                                 <1>       ; by the LGPL VGABios developers Team (2001-2008)
10096                                 <1>       ; 'vgabios.c', 'biosfn_load_gfx_8_16_chars'
10097                                 <1>
10098                                 <1>       ; biosfn_load_gfx_8_16_chars (BL)
10099                                 <1>       ; /* set 0x43 INT pointer */
10100                                 <1>       ; write_word(0x0, 0x43*4, &vgafont16);
10101                                 <1>       ; write_word(0x0, 0x43*4+2, 0xC000);
10102 000034DA C705[528D0100]-        <1>       mov    dword [VGA_INT43H], vgafont16
10102 000034E0 [3C6D0100]             <1>
10103                                 <1>
10104                                 <1>       ; BL    screen rows code: 00H = user-specified (in DL)
10105                                 <1>         ;                       01H = 14 rows
10106                                 <1>         ;                       02H = 25 rows
10107                                 <1>         ;                       03H = 43 rows
10108                                 <1>         ; DL    (when BL=0) custom number of char rows on screen
10109                                 <1>
10110                                 <1>       ; switch (BL) {
10111                                 <1>       ; case 0:
10112                                 <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, DL-1);
10113                                 <1>       ;    break;
10114 000034E4 20DB                  <1>       and    bl, bl
10115 000034E6 7508                  <1>       jnz    short l_gfx_8_16c_1
10116 000034E8 8815[C26A0000]        <1>       mov    [VGA_ROWS], dl  ; not DL-1 !
10117 000034EE EB23                  <1>       jmp    short l_gfx_8_16c_4
10118                                 <1> l_gfx_8_16c_1:
10119                                 <1>       ; case 1:
10120                                 <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
10121                                 <1>       ;    break;
10122 000034F0 FECB                  <1>       dec    bl
10123 000034F2 7509                  <1>       jnz    short l_gfx_8_16c_2
10124                                 <1>       ; bl = 1
10125 000034F4 C605[C26A0000]0E      <1>       mov    byte [VGA_ROWS], 14  ; not 13 !
10126 000034FB EB16                  <1>       jmp    short l_gfx_8_16c_4
10127                                 <1> l_gfx_8_16c_2:
10128 000034FD FECB                  <1>       dec    bl
10129 000034FF 740B                  <1>       jz     short l_gfx_8_16c_3 ; bl = 2
10130 00003501 FECB                  <1>       dec    bl
10131 00003503 750E                  <1>       jnz    short l_gfx_8_16c_4 ; bl > 3
10132                                 <1>       ; bl = 3
10133                                 <1>       ; case 3:
10134                                 <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
```

```
10135                                   <1>       ;    break;
10136 00003505 C605[C26A0000]2B           <1>       mov   byte [VGA_ROWS], 43  ; not 42 !
10137                                   <1> l_gfx_8_16c_3:
10138                                   <1>       ; case 2:
10139                                   <1>       ; default:
10140                                   <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
10141                                   <1>       ;    break;
10142                                   <1>       ; bl = 2 or bl > 3
10143 0000350C C605[C26A0000]19           <1>       mov   byte [VGA_ROWS], 25  ; not 24 !
10144                                   <1>       ; }
10145                                   <1> l_gfx_8_16c_4:
10146                                   <1>       ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, 16);
10147 00003513 C605[BE6A0000]10           <1>         mov     byte [CHAR_HEIGHT], 16
10148                                   <1>       ; }
10149 0000351A C3                       <1>       retn
10150                                   <1>
10151                                   <1> get_font_info:
10152                                   <1>       ; 08/01/2021 (TRDOS 386 v2.0.3)
10153                                   <1>       ; 19/09/2016
10154                                   <1>       ; 08/08/2016
10155                                   <1>       ; 10/07/2016
10156                                   <1>       ; Get Current Character Generator Info (VGA)
10157                                   <1>       ;
10158                                   <1>       ; derived from 'Plex86/Bochs VGABios' source code
10159                                   <1>       ; vgabios-0.7a (2011)
10160                                   <1>       ; by the LGPL VGABios developers Team (2001-2008)
10161                                   <1>       ; 'vgabios.c', 'biosfn_get_font_info'
10162                                   <1>
10163                                   <1>       ; Modified for TRDOS 386 !
10164                                   <1>       ;
10165                                   <1>       ; INPUT ->
10166                                   <1>       ;    AX = 1130h
10167                                   <1>       ;    BL = 0 -> Get info for current VGA font
10168                                   <1>       ;            (BH = unused)
10169                                   <1>       ;    19/09/2016
10170                                   <1>       ;    BL > 0 -> Get requested character font data
10171                                   <1>       ;       BL = 1 -> vgafont8
10172                                   <1>       ;        BL = 2 -> vgafont14
10173                                   <1>       ;       BL = 3 -> vgafont16
10174                                   <1>       ;     ;08/01/2021
10175                                   <1>       ;       BL = 4 -> user defined 8x8 font
10176                                   <1>       ;       BL = 5 -> user defined 8x14 font
10177                                   <1>       ;       BL = 6 -> user defined 8x16 font
10178                                   <1>       ;       BL > 6 -> Invalid function (for now!)
10179                                   <1>       ;       BH = ASCII code of the first character
10180                                   <1>       ;       ECX = Number of characters from the 1st char
10181                                   <1>       ;       ECX >= 256 -> All (256-BH) characters
10182                                   <1>       ;       ECX = 0 -> All characters (BH = unused)
10183                                   <1>       ;        EDX = User's Buffer Address
10184                                   <1>       ; OUTPUT ->
10185                                   <1>       ;    AL = height (scanlines), bytes per character
10186                                   <1>       ;    AH = screen rows
10187                                   <1>       ;    Byte 16-23 of EAX = number of columns
10188                                   <1>       ;    Byte 24-31 of EAX =
10189                                   <1>       ;        0 -> default font (not configured yet)
10190                                   <1>       ;        0FFh -> user defined font
10191                                   <1>       ;        14 = vgafont14
10192                                   <1>       ;         8 = vgafont8
10193                                   <1>       ;        16 = vgafont16
10194                                   <1>       ;   If BL input > 0 ->
10195                                   <1>       ;       EAX = Actual transfer count
10196                                   <1>       ;
10197 0000351B 20DB                     <1>       and   bl, bl
10198 0000351D 740F                     <1>       jz    short gfi_1
10199                                   <1>       ; invalid function (input)
10200                                   <1>       ; 08/01/2021
10201 0000351F 80FB04                   <1>       cmp   bl, 4
10202 00003522 7263                     <1>       jb    short gfi_5
10203 00003524 7441                     <1>       je    short gfi_3
10204 00003526 80FB06                   <1>       cmp   bl, 6
10205 00003529 744C                     <1>       je    short gfi_4
10206                                   <1>       ; bh = 5 or bh > 6
10207                                   <1> gfi_0:
10208 0000352B 31C0                     <1>       xor   eax, eax ; 0
10209 0000352D C3                       <1>       retn
10210                                   <1> gfi_1:
10211 0000352E A0[BE6A0000]             <1>       mov   al, [CHAR_HEIGHT]
10212 00003533 8A25[C26A0000]           <1>       mov   ah, [VGA_ROWS]
10213 00003539 C1E010                   <1>       shl   eax, 16
10214 0000353C A0[BC6A0000]             <1>       mov   al, [CRT_COLS]
10215 00003541 8B0D[528D0100]           <1>       mov   ecx, [VGA_INT43H]
10216 00003547 21C9                     <1>       and   ecx, ecx
10217 00003549 7418                     <1>       jz    short gfi_2 ; 0 = default font
10218                                   <1>       ; 08/01/2021
10219 0000354B FECC                     <1>       dec   ah ; 0FFh
10220 0000354D 81F900400900             <1>       cmp   ecx, VGAFONT16USER
10221 00003553 740E                     <1>       je    short gfi_2
10222 00003555 81F900500900             <1>       cmp   ecx, VGAFONT8USER
10223 0000355B 7406                     <1>       je    short gfi_2
10224 0000355D 8A25[BE6A0000]           <1>       mov   ah, [CHAR_HEIGHT] ; font size = height
10225                                   <1> gfi_2:
10226 00003563 C1C010                   <1>       rol   eax, 16
10227 00003566 C3                       <1>       retn
10228                                   <1> gfi_3:
10229                                   <1>       ; 08/01/2021
10230 00003567 F605[66120300]08         <1>       test  byte [ufont], 08h ; 8x8 user font
10231 0000356E 74BB                     <1>       jz    short gfi_0 ; not loaded !
10232 00003570 BE00500900               <1>       mov   esi, VGAFONT8USER ; *
10233                                   <1>       ;mov  bl, 8
10234                                   <1>       ;jmp  short gfi_8
10235 00003575 EB4D                     <1>       jmp   short gfi_10
10236                                   <1> gfi_4:
10237                                   <1>       ; 08/01/2021
10238 00003577 F605[66120300]10         <1>       test  byte [ufont], 10h ; 8x16 user font
10239 0000357E 74AB                     <1>       jz    short gfi_0 ; not loaded !
```

```
10240 00003580 BE00400900          <1>       mov    esi, VGAFONT16USER ; *
10241 00003585 EB15                <1>       jmp    short gfi_7
10242                              <1> gfi_5:
10243 00003587 80FB02              <1>       cmp    bl, 2
10244 0000358A 7233                <1>       jb     short gfi_9
10245 0000358C 7709                <1>       ja     short gfi_6
10246                              <1>       ;BL = 2 -> vgafont14
10247 0000358E BE[3C5F0100]        <1>       mov    esi, vgafont14 ; *
10248 00003593 B30E                <1>       mov    bl, 14
10249 00003595 EB07                <1>       jmp    short gfi_8
10250                              <1> gfi_6:
10251                              <1>       ;BL = 3 -> vgafont16
10252 00003597 BE[3C6D0100]        <1>       mov    esi, vgafont16 ; *
10253                              <1> gfi_7:
10254 0000359C B310                <1>       mov    bl, 16
10255                              <1> gfi_8:
10256 0000359E 89D7                <1>       mov    edi, edx ; **
10257 000035A0 09C9                <1>       or     ecx, ecx
10258 000035A2 7424                <1>       jz     short gfi_11 ; all chars from the 00h
10259 000035A4 88F8                <1>       mov    al, bh ; character index
10260 000035A6 F6E3                <1>       mul    bl ; char index * char height/size
10261 000035A8 0FB7D0              <1>       movzx  edx, ax
10262 000035AB 01D6                <1>       add    esi, edx ; *
10263 000035AD 66BAFF00            <1>       mov    dx, 255
10264 000035B1 28FA                <1>       sub    dl, bh
10265 000035B3 6642                <1>       inc    dx
10266 000035B5 39D1                <1>       cmp    ecx, edx
10267 000035B7 770F                <1>       ja     short gfi_11
10268 000035B9 7412                <1>       je     short gfi_12
10269 000035BB 89D1                <1>       mov    ecx, edx
10270 000035BD EB0E                <1>       jmp    short gfi_12
10271                              <1> gfi_9:
10272                              <1>       ;BL = 1 -> vgafont8
10273 000035BF BE[3C570100]        <1>       mov    esi, vgafont8 ; *
10274                              <1> gfi_10:
10275 000035C4 B308                <1>       mov    bl, 8
10276 000035C6 EBD6                <1>       jmp    short gfi_8
10277                              <1> gfi_11:
10278 000035C8 B900010000          <1>       mov    ecx, 256
10279                              <1> gfi_12:
10280                              <1>       ; 08/01/2021
10281 000035CD 89C8                <1>       mov    eax, ecx ; character count
10282 000035CF 30FF                <1>       xor    bh, bh
10283 000035D1 66F7E3              <1>       mul    bx ; char count * char height/size
10284 000035D4 89C1                <1>       mov    ecx, eax
10285                              <1>
10286                              <1>       ; ESI = source address in system space
10287                              <1>       ; EDI = user's buffer address
10288                              <1>       ; ECX = transfer (byte) count
10289 000035D6 E867DF0000          <1>       call   transfer_to_user_buffer
10290 000035DB 89C8                <1>       mov    eax, ecx ; actual transfer count
10291 000035DD C3                  <1>       retn
10292                              <1>
10293                              <1> vga_pal_funcs:
10294                              <1>       ; 10/08/2016
10295                              <1>       ; VGA Palette functions
10296                              <1>       ;
10297                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
10298                              <1>       ; vgabios-0.7a (2011)
10299                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
10300                              <1>       ; 'vgabios.c', 'vgarom.asm'
10301                              <1>
10302 000035DE 3C00                <1>       cmp    al, 0
10303 000035E0 0F848F000000        <1>         je      set_single_palette_reg
10304                              <1> vga_palf_1001:
10305 000035E6 3C01                <1>       cmp    al, 1
10306 000035E8 0F84B0000000        <1>         je      set_overscan_border_color
10307                              <1> vga_palf_1002:
10308 000035EE 3C02                <1>       cmp    al, 2
10309 000035F0 0F84AC000000        <1>         je      set_all_palette_reg
10310                              <1> vga_palf_1003:
10311 000035F6 3C03                <1>       cmp    al, 3
10312 000035F8 0F84E4000000        <1>         je      toggle_intensity
10313                              <1> vga_palf_1007:
10314 000035FE 3C07                <1>       cmp    al, 7
10315 00003600 0F8409010000        <1>         je      get_single_palette_reg
10316 00003606 7266                <1>       jb     short vga_palf_unknown
10317                              <1> vga_palf_1008:
10318 00003608 3C08                <1>       cmp    al, 8
10319 0000360A 0F8433010000        <1>         je      read_overscan_border_color
10320                              <1> vga_palf_1009:
10321 00003610 3C09                <1>       cmp    al, 9
10322 00003612 0F842F010000        <1>       je     get_all_palette_reg
10323                              <1> vga_palf_1010:
10324 00003618 3C10                <1>       cmp    al, 10h
10325 0000361A 0F8483010000        <1>       je     set_single_dac_reg
10326 00003620 724C                <1>       jb     short vga_palf_unknown
10327                              <1> vga_palf_1012:
10328 00003622 3C12                <1>       cmp    al, 12h
10329 00003624 0F8492010000        <1>         je      set_all_dac_reg
10330 0000362A 7242                <1>       jb     short vga_palf_unknown
10331                              <1> vga_palf_1013:
10332 0000362C 3C13                <1>       cmp    al, 13h
10333 0000362E 0F84C6010000        <1>         je      select_video_dac_color_page
10334                              <1> vga_palf_1015:
10335 00003634 3C15                <1>       cmp    al, 15h
10336 00003636 0F840A020000        <1>       je     read_single_dac_reg
10337 0000363C 7230                <1>       jb     short vga_palf_unknown
10338                              <1> vga_palf_1017:
10339 0000363E 3C17                <1>       cmp    al, 17h
10340 00003640 0F8420020000        <1>         je      read_all_dac_reg
10341 00003646 7226                <1>       jb     short vga_palf_unknown
10342                              <1> vga_palf_1018:
10343 00003648 3C18                <1>       cmp    al, 18h
10344 0000364A 0F8456020000        <1>         je      set_pel_mask
```

```
10345                           <1> vga_palf_1019:
10346 00003650 3C19             <1>         cmp    al, 19h
10347 00003652 0F845A020000     <1>           je      read_pel_mask
10348                           <1> vga_palf_101A:
10349 00003658 3C1A             <1>         cmp    al, 1Ah
10350 0000365A 0F8460020000     <1>           je      read_video_dac_state
10351                           <1> vga_palf_101B:
10352 00003660 3C1B             <1>         cmp    al, 1Bh
10353                           <1>         ;jne   short vga_palf_unknown
10354 00003662 770A             <1>         ja     short vga_palf_unknown
10355                           <1>
10356 00003664 E8CDF5FFFF       <1>         call   gray_scale_summing
10357 00003669 E97BE4FFFF       <1>           jmp     VIDEO_RETURN
10358                           <1>
10359                           <1> vga_palf_unknown:
10360 0000366E 29C0             <1>         sub    eax, eax ; 0 = invalid function
10361 00003670 E979E4FFFF       <1>           jmp     _video_return
10362                           <1>
10363                           <1> set_single_palette_reg:
10364                           <1>         ; 12/04/2021 (TRDOS 386 v2.0.3, 32 bit push/pop)
10365                           <1>         ; 10/08/2016
10366                           <1>         ; Set One Palette Register
10367                           <1>         ; BL = register number to set
10368                           <1>         ;     (a 4-bit attribute nibble: 00h-0Fh)
10369                           <1>         ; BH = 6-bit RGB color to display
10370                           <1>         ;      for that attribute
10371                           <1>
10372 00003675 80FB14             <1>         cmp    bl, 14h
10373                           <1>         ;ja    short no_actl_reg1
10374 00003678 0F876BE4FFFF     <1>         ja     VIDEO_RETURN
10375                           <1>         ;push  ax
10376                           <1>         ;push  dx
10377                           <1>         ; 12/04/2021
10378 0000367E 50               <1>         push   eax
10379 0000367F 52               <1>         push   edx
10380 00003680 66BADA03         <1>         mov    dx, 3DAh ; VGAREG_ACTL_RESET
10381 00003684 EC               <1>         in     al, dx
10382 00003685 66BAC003         <1>         mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10383 00003689 88D8             <1>         mov    al, bl
10384 0000368B EE               <1>         out    dx, al
10385 0000368C 88F8             <1>         mov    al, bh
10386 0000368E EE               <1>         out    dx, al
10387 0000368F B020             <1>         mov    al, 20h
10388 00003691 EE               <1>         out    dx, al
10389                           <1>         ; ifdef VBOX
10390 00003692 66BADA03         <1>         mov    dx, 3DAh ; VGAREG_ACTL_RESET
10391 00003696 EC               <1>         in     al, dx
10392                           <1>         ; endif ; VBOX
10393                           <1>         ;pop   dx
10394                           <1>         ;pop   ax
10395                           <1>         ; 12/04/2021
10396 00003697 5A               <1>         pop    edx
10397 00003698 58               <1>         pop    eax
10398                           <1> ;no_actl_reg1:
10399 00003699 E94BE4FFFF       <1>         jmp    VIDEO_RETURN
10400                           <1>
10401                           <1> set_overscan_border_color:
10402                           <1>         ; 10/08/2016
10403                           <1>         ; Set Overscan/Border Color Register
10404                           <1>         ; BH = 6-bit RGB color to display
10405                           <1>         ;      for that attribute
10406                           <1>
10407 0000369E B311             <1>         mov    bl, 11h
10408 000036A0 EBD3             <1>         jmp    short set_single_palette_reg
10409                           <1>
10410                           <1> set_all_palette_reg:
10411                           <1>         ; 10/08/2016
10412                           <1>         ; Set All Palette Registers and Overscan
10413                           <1>         ; EDX = Address of 17 bytes;
10414                           <1>         ; an rgbRGB value for each of 16 palette
10415                           <1>         ; registers plus one for the border.
10416                           <1>
10417 000036A2 89D6             <1>         mov    esi, edx ; user buffer
10418 000036A4 B911000000       <1>         mov    ecx, 17
10419 000036A9 89E7             <1>         mov    edi, esp
10420 000036AB 83EC14           <1>         sub    esp, 20
10421 000036AE E8D9DE0000       <1>         call   transfer_from_user_buffer
10422                           <1>         ;jc    VIDEO_RETURN
10423                           <1>
10424 000036B3 66BADA03         <1>         mov    dx, 3DAh ; VGAREG_ACTL_RESET
10425 000036B7 EC               <1>         in     al, dx
10426 000036B8 B100             <1>         mov    cl, 0
10427 000036BA 66BAC003         <1>         mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10428                           <1> set_palette_loop:
10429 000036BE 88C8             <1>         mov    al, cl
10430 000036C0 EE               <1>         out    dx, al
10431 000036C1 8A07             <1>         mov    al, [edi]
10432 000036C3 EE               <1>         out    dx, al
10433 000036C4 47               <1>         inc    edi
10434 000036C5 FEC1             <1>         inc    cl
10435 000036C7 80F910           <1>         cmp    cl, 10h
10436 000036CA 75F2             <1>         jne    short set_palette_loop
10437 000036CC B011             <1>         mov    al, 11h
10438 000036CE EE               <1>         out    dx, al
10439 000036CF 8A07             <1>         mov    al, [edi]
10440 000036D1 EE               <1>         out    dx, al
10441 000036D2 B020             <1>         mov    al, 20h
10442 000036D4 EE               <1>         out    dx, al
10443                           <1>         ; ifdef VBOX
10444 000036D5 66BADA03         <1>         mov    dx, 3DAh ; VGAREG_ACTL_RESET
10445 000036D9 EC               <1>         in     al, dx
10446                           <1>         ; endif ; VBOX
10447 000036DA 83C414           <1>         add    esp, 20
10448 000036DD E907E4FFFF       <1>         jmp    VIDEO_RETURN
10449                           <1>
```

```
                                <1> toggle_intensity:
                                <1>       ; 10/08/2016
                                <1>       ; Select Foreground Blink or Bold Background
                                <1>       ; BL = 00h = enable bold backgrounds
                                <1>       ;          (16 background colors)
                                <1>         ;      01h = enable blinking foreground
                                <1>       ;          (8 background colors)
                                <1>
000036E2 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
000036E6 EC                     <1>       in     al, dx
000036E7 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
000036EB B010                   <1>       mov    al, 10h
000036ED EE                     <1>       out    dx, al
000036EE 66BAC103               <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
000036F2 EC                     <1>       in     al, dx
000036F3 24F7                   <1>       and    al, 0F7h
000036F5 80E301                 <1>       and    bl, 01h
000036F8 C0E303                 <1>       shl    bl, 3
000036FB 08D8                   <1>       or     al, bl
000036FD 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
00003701 EE                     <1>       out    dx, al
00003702 B020                   <1>       mov    al, 20h
00003704 EE                     <1>       out    dx, al
                                <1>       ; ifdef VBOX
00003705 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
00003709 EC                     <1>       in     al, dx
                                <1>       ; endif ; VBOX
0000370A E9DAE3FFFF             <1>       jmp    VIDEO_RETURN
                                <1>
                                <1> get_single_palette_reg:
                                <1>       ; 10/08/2016
                                <1>       ; Read One Palette Register
                                <1>         ; INPUT:
                                <1>       ; BL = Palette register to read (00h-0Fh)
                                <1>       ; OUTPUT:
                                <1>       ; BH = Current rgbRGB value of specified register
                                <1>       ;      for that attribute
                                <1>
0000370F 80FB14                 <1>       cmp    bl, 14h
                                <1>       ;ja    short no_actl_reg2
00003712 0F87D1E3FFFF           <1>       ja     VIDEO_RETURN
                                <1>
00003718 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
0000371C EC                     <1>       in     al, dx
0000371D 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
00003721 88D8                   <1>       mov    al, bl
00003723 EE                     <1>       out    dx, al
00003724 66BAC103               <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
00003728 EC                     <1>       in     al, dx
00003729 8844240D               <1>       mov    [esp+13], al ; bh
0000372D 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
00003731 EC                     <1>       in     al, dx
00003732 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
00003736 B020                   <1>       mov    al, 20h
00003738 EE                     <1>       out    dx, al
                                <1>       ; ifdef VBOX
00003739 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
0000373D EC                     <1>       in     al, dx
                                <1>       ; endif ; VBOX
0000373E E9A6E3FFFF             <1>       jmp    VIDEO_RETURN
                                <1>
                                <1> read_overscan_border_color:
                                <1>       ; 10/08/2016
                                <1>       ; Read Overscan Register
                                <1>       ; OUTPUT:
                                <1>       ; BH = current rgbRGB value
                                <1>       ;      of the overscan/border register
                                <1>
00003743 B311                   <1>       mov    bl, 11h
00003745 EBC8                   <1>       jmp    short get_single_palette_reg
                                <1>
                                <1> get_all_palette_reg:
                                <1>       ; 10/08/2016
                                <1>       ; Read All Palette Registers
                                <1>       ; EDX = Address of 17-byte buffer
                                <1>       ;      to receive data
                                <1>
00003747 89D7                   <1>       mov    edi, edx
00003749 89E3                   <1>       mov    ebx, esp
0000374B 89DE                   <1>       mov    esi, ebx
0000374D 83EC14                 <1>       sub    esp, 20
                                <1>
00003750 B100                   <1>       mov    cl, 0
                                <1> get_palette_loop:
00003752 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
00003756 EC                     <1>       in     al, dx
00003757 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
0000375B 88C8                   <1>       mov    al, cl
0000375D EE                     <1>       out    dx, al
0000375E 66BAC103               <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
00003762 EC                     <1>       in     al, dx
00003763 8803                   <1>       mov    [ebx], al
00003765 43                     <1>       inc    ebx
00003766 FEC1                   <1>       inc    cl
00003768 80F910                 <1>       cmp    cl, 10h
0000376B 75E5                   <1>       jne    short get_palette_loop
0000376D 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
00003771 EC                     <1>       in     al, dx
00003772 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
00003776 B011                   <1>       mov    al, 11h
00003778 EE                     <1>       out    dx, al
00003779 66BAC103               <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
0000377D EC                     <1>       in     al, dx
0000377E 8803                   <1>       mov    [ebx], al
00003780 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
```

```
10555 00003784 EC                <1>       in     al, dx
10556 00003785 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10557 00003789 B020              <1>       mov    al, 20h
10558 0000378B EE                <1>       out    dx, al
10559                            <1>       ; ifdef VBOX
10560 0000378C 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10561 00003790 EC                <1>       in     al, dx
10562                            <1>       ; endif ; VBOX
10563                            <1>
10564 00003791 B911000000        <1>       mov    ecx, 17 ; transfer (byte) count
10565                            <1>       ; ESI = source address in system space
10566                            <1>       ; EDI = user's buffer address
10567 00003796 E8A7DD0000        <1>       call   transfer_to_user_buffer
10568                            <1>
10569 0000379B 83C414            <1>       add    esp, 20
10570 0000379E E946E3FFFF        <1>       jmp    VIDEO_RETURN
10571                            <1>
10572                            <1> set_single_dac_reg:
10573                            <1>       ; 12/04/2021 (TRDOS 386 v2.0.3, 32 bit push/pop)
10574                            <1>       ; 10/08/2016
10575                            <1>       ; Set One DAC Color Register
10576                            <1>       ; BX = color register to set (0-255)
10577                            <1>         ; CH = green value (00h-3Fh)
10578                            <1>         ; CL = blue value  (00h-3Fh)
10579                            <1>         ; DH = red value   (00h-3Fh)
10580                            <1>
10581                            <1>       ;push  dx
10582                            <1>       ; 12/04/2021
10583 000037A3 52                <1>       push   edx
10584 000037A4 66BAC803          <1>       mov    dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
10585 000037A8 88D8              <1>       mov    al, bl
10586 000037AA EE                <1>       out    dx, al
10587                            <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
10588 000037AB 6642              <1>       inc    dx
10589                            <1>       ;pop   ax
10590                            <1>       ; 12/04/2021
10591 000037AD 58                <1>       pop    eax
10592 000037AE 88E0              <1>       mov    al, ah
10593 000037B0 EE                <1>       out    dx, al
10594 000037B1 88E8              <1>       mov    al, ch
10595 000037B3 EE                <1>       out    dx, al
10596 000037B4 88C8              <1>       mov    al, cl
10597 000037B6 EE                <1>       out    dx, al
10598 000037B7 E92DE3FFFF        <1>       jmp    VIDEO_RETURN
10599                            <1>
10600                            <1> set_all_dac_reg:
10601                            <1>       ; 12/08/2016
10602                            <1>       ; 11/08/2016
10603                            <1>       ; 10/08/2016
10604                            <1>       ; Set a Block of DAC Color Register
10605                            <1>       ; BX = first DAC register to set (0-00FFh)
10606                            <1>       ; ECX = number of registers to set (0-00FFh)
10607                            <1>       ; EDX = addr of a table of R,G,B values
10608                            <1>       ;      (it will be CX*3 bytes long)
10609                            <1>
10610 000037BC 89D6              <1>       mov    esi, edx ; user buffer
10611 000037BE 89CA              <1>       mov    edx, ecx
10612 000037C0 66D1E1            <1>       shl    cx, 1 ; *2
10613 000037C3 01D1              <1>       add    ecx, edx ; ecx = 3*ecx
10614 000037C5 89E5              <1>       mov    ebp, esp
10615 000037C7 89EF              <1>       mov    edi, ebp
10616 000037C9 29CF              <1>       sub    edi, ecx
10617 000037CB 6683E7FC          <1>       and    di, 0FFFCh ; (dword alignment)
10618 000037CF 89FC              <1>       mov    esp, edi
10619 000037D1 E8B6DD0000        <1>       call   transfer_from_user_buffer
10620                            <1>       ;jc    VIDEO_RETURN
10621                            <1>
10622 000037D6 89D1              <1>       mov    ecx, edx
10623 000037D8 66BAC803          <1>       mov    dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
10624 000037DC 88D8              <1>       mov    al, bl
10625 000037DE EE                <1>       out    dx, al
10626 000037DF 66BAC903          <1>       mov    dx, 3C9h ; VGAREG_DAC_DATA
10627                            <1> set_dac_loop:
10628 000037E3 8A07              <1>       mov    al, [edi]
10629 000037E5 EE                <1>       out    dx, al
10630 000037E6 47                <1>       inc    edi
10631 000037E7 8A07              <1>       mov    al, [edi]
10632 000037E9 EE                <1>       out    dx, al
10633 000037EA 47                <1>       inc    edi
10634 000037EB 8A07              <1>       mov    al, [edi]
10635 000037ED EE                <1>       out    dx, al
10636 000037EE 47                <1>       inc    edi
10637 000037EF 6649              <1>       dec    cx
10638 000037F1 75F0              <1>       jnz    short set_dac_loop
10639 000037F3 89EC              <1>       mov    esp, ebp
10640 000037F5 E9EFE2FFFF        <1>       jmp    VIDEO_RETURN
10641                            <1>
10642                            <1> select_video_dac_color_page:
10643                            <1>       ; 12/04/2021 (TRDOS 386 v2.0.3, 32 bit push/pop)
10644                            <1>       ; 10/08/2016
10645                            <1>       ; DAC Color Paging Functions
10646                            <1>       ; BL = 00H = select color paging mode
10647                            <1>         ;       BH = paging mode
10648                            <1>         ;            00h = 4 blocks of 64 registers
10649                            <1>         ;            01h = 16 blocks of 16 registers
10650                            <1>       ; BL = 01H = activate color page
10651                            <1>         ;       BH = DAC color page number
10652                            <1>         ;            00h-03h (4-page/64-reg mode)
10653                            <1>         ;            00h-0Fh (16-page/16-reg mode)
10654                            <1>
10655 000037FA 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10656 000037FE EC                <1>       in     al, dx
10657 000037FF 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10658 00003803 B010              <1>       mov    al, 10h
10659 00003805 EE                <1>       out    dx, al
```

```
10660 00003806 66BAC103           <1>       mov   dx, 3C1h ; VGAREG_ACTL_READ_DATA
10661 0000380A EC                 <1>       in    al, dx
10662 0000380B 80E301             <1>       and   bl, 01h
10663 0000380E 750E               <1>       jnz   short set_dac_page
10664 00003810 247F               <1>       and   al, 07Fh
10665 00003812 C0E707             <1>       shl   bh, 7
10666 00003815 08F8               <1>       or    al, bh
10667 00003817 66BAC003           <1>       mov   dx, 3C0h ; VGAREG_ACTL_ADDRESS
10668 0000381B EE                 <1>       out   dx, al
10669 0000381C EB1B               <1>       jmp   short set_actl_normal
10670                             <1> set_dac_page:
10671                             <1>       ;push  ax
10672                             <1>       ; 12/04/2021
10673 0000381E 50                 <1>       push  eax
10674 0000381F 66BADA03           <1>       mov   dx, 3DAh ; VGAREG_ACTL_RESET
10675 00003823 EC                 <1>       in    al, dx
10676 00003824 66BAC003           <1>       mov   dx, 3C0h ; VGAREG_ACTL_ADDRESS
10677 00003828 B014               <1>       mov   al, 14h
10678 0000382A EE                 <1>       out   dx, al
10679                             <1>       ;pop   ax
10680                             <1>       ; 12/04/2021
10681 0000382B 58                 <1>       pop   eax
10682 0000382C 2480               <1>       and   al, 80h
10683 0000382E 7503               <1>       jnz   short set_dac_16_page
10684 00003830 C0E702             <1>       shl   bh, 2
10685                             <1> set_dac_16_page:
10686 00003833 80E70F             <1>       and   bh, 0Fh
10687 00003836 88F8               <1>       mov   al, bh
10688 00003838 EE                 <1>       out   dx, al
10689                             <1> set_actl_normal:
10690 00003839 B020               <1>       mov   al, 20h
10691 0000383B EE                 <1>       out   dx, al
10692                             <1>       ; ifdef VBOX
10693 0000383C 66BADA03           <1>       mov   dx, 3DAh ; VGAREG_ACTL_RESET
10694 00003840 EC                 <1>       in    al, dx
10695                             <1>       ; endif ; VBOX
10696 00003841 E9A3E2FFFF         <1>       jmp   VIDEO_RETURN
10697                             <1>
10698                             <1> read_single_dac_reg:
10699                             <1>       ; 10/08/2016
10700                             <1>       ; Read One DAC Color Register
10701                             <1>       ; INPUT:
10702                             <1>       ; BX = color register to read (0-255)
10703                             <1>       ; OUTPUT:
10704                             <1>       ; CH = green value (00h-3Fh)
10705                             <1>         ; CL = blue value  (00h-3Fh)
10706                             <1>         ; DH = red value   (00h-3Fh)
10707                             <1>
10708 00003846 66BAC703           <1>       mov   dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
10709 0000384A 88D8               <1>       mov   al, bl
10710 0000384C EE                 <1>       out   dx, al
10711 0000384D 66BAC903           <1>       mov   dx, 3C9h ; VGAREG_DAC_DATA
10712 00003851 EC                 <1>       in    al, dx
10713 00003852 88442415           <1>       mov   [esp+21], al ; dh
10714 00003856 EC                 <1>       in    al, dx
10715 00003857 88C5               <1>       mov   ch, al
10716 00003859 EC                 <1>       in    al, dx
10717 0000385A 88C1               <1>       mov   cl, al
10718 0000385C 66894C2410         <1>       mov   [esp+16], cx ; cx
10719 00003861 E983E2FFFF         <1>       jmp   VIDEO_RETURN
10720                             <1>
10721                             <1> read_all_dac_reg:
10722                             <1>       ; 12/08/2016
10723                             <1>       ; 11/08/2016
10724                             <1>       ; 10/08/2016
10725                             <1>       ; Read a Block of DAC Color Registers
10726                             <1>         ; BX = first DAC register to read (0-00FFh)
10727                             <1>         ; ECX = number of registers to read (0-00FFh)
10728                             <1>         ; EDX = addr of a buffer to hold R,G,B values
10729                             <1>       ;      (CX*3 bytes long)
10730                             <1>
10731 00003866 89D7               <1>       mov   edi, edx ; user buffer
10732 00003868 89CA               <1>       mov   edx, ecx
10733 0000386A 66D1E2             <1>       shl   dx, 1 ; *2
10734 0000386D 01CA               <1>       add   edx, ecx ; edx = 3*ecx
10735 0000386F 89E5               <1>       mov   ebp, esp
10736 00003871 89EE               <1>       mov   esi, ebp
10737 00003873 29D6               <1>       sub   esi, edx
10738 00003875 6683E6FC           <1>       and   si, 0FFFCh ; (dword alignment)
10739 00003879 89F4               <1>       mov   esp, esi
10740 0000387B 52                 <1>       push  edx ; 3*ecx
10741 0000387C 66BAC703           <1>       mov   dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
10742 00003880 88D8               <1>       mov   al, bl
10743 00003882 EE                 <1>       out   dx, al
10744 00003883 66BAC903           <1>       mov   dx, 3C9h ; VGAREG_DAC_DATA
10745 00003887 89F3               <1>       mov   ebx, esi
10746                             <1> read_dac_loop:
10747 00003889 EC                 <1>       in    al, dx
10748 0000388A 8803               <1>       mov   [ebx], al
10749 0000388C 43                 <1>       inc   ebx
10750 0000388D EC                 <1>       in    al, dx
10751 0000388E 8803               <1>       mov   [ebx], al
10752 00003890 43                 <1>       inc   ebx
10753 00003891 EC                 <1>       in    al, dx
10754 00003892 8803               <1>       mov   [ebx], al
10755 00003894 43                 <1>       inc   ebx
10756 00003895 6649               <1>       dec   cx
10757 00003897 75F0               <1>       jnz   short read_dac_loop
10758 00003899 59                 <1>       pop   ecx ; 3*ecx
10759                             <1>       ; ECX = transfer (byte) count
10760                             <1>       ; ESI = source address in system space
10761                             <1>       ; EDI = user's buffer address
10762 0000389A E8A3DC0000         <1>       call  transfer_to_user_buffer
10763 0000389F 89EC               <1>       mov   esp, ebp
10764 000038A1 E943E2FFFF         <1>       jmp   VIDEO_RETURN
```

```
                                <1>
                                <1> set_pel_mask:
                                <1>       ; 10/08/2016
                                <1>       ; BL = mask value
000038A6 66BAC603               <1>       mov    dx, 3C6h ; VGAREG_PEL_MASK
000038AA 88D8                   <1>       mov    al, bl
000038AC EE                     <1>       out    dx, al
000038AD E937E2FFFF             <1>       jmp    VIDEO_RETURN
                                <1>
                                <1> read_pel_mask:
                                <1>       ; 10/08/2016
                                <1>       ; Output: BL = mask value
000038B2 66BAC603               <1>       mov    dx, 3C6h ; VGAREG_PEL_MASK
000038B6 EC                     <1>       in     al, dx
000038B7 8844240C               <1>       mov    [esp+12], al ; bl
000038BB E929E2FFFF             <1>       jmp    VIDEO_RETURN
                                <1>
                                <1> read_video_dac_state:
                                <1>       ; 10/08/2016
                                <1>       ; Query DAC Color Paging State
                                <1>       ; Output:
                                <1>       ; BH = current active DAC color page
                                <1>         ; BL = current active DAC paging mode
                                <1>
000038C0 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
000038C4 EC                     <1>       in     al, dx
000038C5 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
000038C9 B010                   <1>       mov    al, 10h
000038CB EE                     <1>       out    dx, al
000038CC 66BAC103               <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
000038D0 EC                     <1>       in     al, dx
000038D1 88C3                   <1>       mov    bl, al
000038D3 C0EB07                 <1>       shr    bl, 7
000038D6 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
000038DA EC                     <1>       in     al, dx
000038DB 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
000038DF B014                   <1>       mov    al, 14h
000038E1 EE                     <1>       out    dx, al
000038E2 66BAC103               <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
000038E6 EC                     <1>       in     al, dx
000038E7 88C7                   <1>       mov    bh, al
000038E9 80E70F                 <1>       and    bh, 0Fh
000038EC F6C301                 <1>       test   bl, 01
000038EF 7503                   <1>       jnz    short get_dac_16_page
000038F1 C0EF02                 <1>       shr    bh, 2
                                <1> get_dac_16_page:
000038F4 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
000038F8 EC                     <1>       in     al, dx
000038F9 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
000038FD B020                   <1>       mov    al, 20h
000038FF EE                     <1>       out    dx, al
                                <1>       ; ifdef VBOX
00003900 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
00003904 EC                     <1>       in     al, dx
                                <1>       ; endif ; VBOX
00003905 66895C240C             <1>       mov    [esp+12], bx ; bx
0000390A E9DAE1FFFF             <1>       jmp    VIDEO_RETURN
                                <1>
                                <1> ; 23/11/2020 - TRDOS 386 v2.0.3
                                <1> ; VBE 2 BOCHS/QEMU emulator extensions
                                <1> ;     for TRDOS 386 v2 kernel (video bios)
                                <1>
                                <1> ; BOCH/QEMU VBE2 VGA BIOS code
                                <1> ;     by Jeroen Janssen (2002)
                                <1> ;     by Volker Rupper (2003-2020)
                                <1> ; vbe.c (02/01/2020)
                                <1>
                                <1> ; vbe.h (02/01/2020)
                                <1>
                                <1> VBE_DISPI_BANK_ADDRESS    equ    0A0000h
                                <1> VBE_DISPI_BANK_SIZE_KB    equ    64
                                <1>
                                <1> VBE_DISPI_MAX_XRES equ    2560
                                <1> VBE_DISPI_MAX_YRES equ    1600
                                <1>
                                <1> VBE_DISPI_IOPORT_INDEX    equ    01CEh
                                <1> VBE_DISPI_IOPORT_DATA     equ    01CFh
                                <1>
                                <1> VBE_DISPI_INDEX_ID equ    00h
                                <1> VBE_DISPI_INDEX_XRES      equ    01h
                                <1> VBE_DISPI_INDEX_YRES      equ    02h
                                <1> VBE_DISPI_INDEX_BPP       equ    03h
                                <1> VBE_DISPI_INDEX_ENABLE    equ    04h
                                <1> VBE_DISPI_INDEX_BANK      equ    05h
                                <1> VBE_DISPI_INDEX_VIRT_WIDTH equ  06h
                                <1> VBE_DISPI_INDEX_VIRT_HEIGHT equ 07h
                                <1> VBE_DISPI_INDEX_X_OFFSET equ    08h
                                <1> VBE_DISPI_INDEX_Y_OFFSET equ    09h
                                <1> VBE_DISPI_INDEX_VIDEO_MEMORY_64K equ 0Ah
                                <1> VBE_DISPI_INDEX_DDC       equ    0Bh
                                <1>
                                <1> VBE_DISPI_ID0             equ    0B0C0h
                                <1> VBE_DISPI_ID1             equ    0B0C1h
                                <1> VBE_DISPI_ID2             equ    0B0C2h
                                <1> VBE_DISPI_ID3             equ    0B0C3h
                                <1> VBE_DISPI_ID4             equ    0B0C4h
                                <1> VBE_DISPI_ID5             equ    0B0C5h
                                <1>
                                <1> VBE_DISPI_DISABLED equ    00h
                                <1> VBE_DISPI_ENABLED  equ    01h
                                <1> VBE_DISPI_GETCAPS  equ    02h
                                <1> VBE_DISPI_8BIT_DAC equ    20h
                                <1> VBE_DISPI_LFB_ENABLED     equ    40h
                                <1> VBE_DISPI_NOCLEARMEM      equ    80h
                                <1>
```

```
                                <1> VBE_DISPI_LFB_PHYSICAL_ADDRESS equ 0E0000000h
                                <1>
                                <1> ; ***
                                <1>
                                <1> ;// VBE Return Status Info
                                <1> ;// AL
                                <1> VBE_RETURN_STATUS_SUPPORTED     equ    4Fh
                                <1> VBE_RETURN_STATUS_UNSUPPORTED   equ    00h
                                <1> ;// AH
                                <1> VBE_RETURN_STATUS_SUCCESSFULL   equ    00h
                                <1> VBE_RETURN_STATUS_FAILED  equ    01h
                                <1> VBE_RETURN_STATUS_NOT_SUPPORTED equ    02h
                                <1> VBE_RETURN_STATUS_INVALID equ    03h
                                <1>
                                <1> ;// VBE Mode Numbers
                                <1>
                                <1> VBE_MODE_VESA_DEFINED           equ    0100h
                                <1> VBE_MODE_REFRESH_RATE_USE_CRTC  equ    0800h
                                <1> VBE_MODE_LINEAR_FRAME_BUFFER    equ    4000h
                                <1> VBE_MODE_PRESERVE_DISPLAY_MEMORY equ   8000h
                                <1>
                                <1> ;// Mode Attributes
                                <1>
                                <1> VBE_MODE_ATTRIBUTE_SUPPORTED              equ 0001h
                                <1> VBE_MODE_ATTRIBUTE_EXTENDED_INFO_AVAILABLE equ      0002h
                                <1> VBE_MODE_ATTRIBUTE_COLOR_MODE             equ 0008h
                                <1> VBE_MODE_ATTRIBUTE_GRAPHICS_MODE   equ 0010h
                                <1> VBE_MODE_ATTRIBUTE_LINEAR_FRAME_BUFFER_MODE equ     0080h
                                <1> VBE_MODE_ATTRIBUTE_DOUBLE_SCAN_MODE       equ 0100h
                                <1> VBE_MODE_ATTRIBUTE_INTERLACE_MODE         equ 0200h
                                <1>
                                <1> ;// Window attributes
                                <1>
                                <1> VBE_WINDOW_ATTRIBUTE_RELOCATABLE equ   01h
                                <1> VBE_WINDOW_ATTRIBUTE_READABLE    equ   02h
                                <1> VBE_WINDOW_ATTRIBUTE_WRITEABLE   equ   04h
                                <1>
                                <1> ;/* Video memory */
                                <1> VGAMEM_GRAPH equ 0A000h
                                <1> VGAMEM_CTEXT equ 0B800h
                                <1> ;VGAMEM_MTEXT equ 0B000h
                                <1>
                                <1> ;// Memory model
                                <1>
                                <1> ;VBE_MEMORYMODEL_TEXT_MODE      equ    00h
                                <1> ;VBE_MEMORYMODEL_CGA_GRAPHICS   equ    01h
                                <1> ;VBE_MEMORYMODEL_PLANAR         equ    03h
                                <1> VBE_MEMORYMODEL_PACKED_PIXEL    equ    04h
                                <1> ;VBE_MEMORYMODEL_NON_CHAIN_4_256 equ   05h
                                <1> VBE_MEMORYMODEL_DIRECT_COLOR    equ    06h
                                <1> ;VBE_MEMORYMODEL_YUV            equ    07h
                                <1>
                                <1> ;// DirectColorModeInfo
                                <1>
                                <1> ;VBE_DIRECTCOLOR_COLOR_RAMP_PROGRAMMABLE equ 01h
                                <1> VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE equ 02h
                                <1>
                                <1> VBE_DISPI_TOTAL_VIDEO_MEMORY_MB equ 16
                                <1>
                                <1> ; 24/11/2020
                                <1> ; vbe.c
                                <1>
                                <1> %if 1
                                <1>
                                <1> _vbe_biosfn_return_mode_info:
                                <1>       ; 15/12/2020
                                <1>       ; 12/12/2020
                                <1>       ; Return VBE Mode Information
                                <1>       ; (call from 'sysvideo')
                                <1>       ;
                                <1>       ; Input:
                                <1>       ;     cx = video (bios) mode
                                <1>       ; Output:
                                <1>       ;     cf = 0 -> (successful)
                                <1>       ;        MODE_INFO_LIST addr contains MODEINFO
                                <1>       ;     cf = 1 -> error
                                <1>       ;
                                <1>       ; Modified registers: eax, edx, edi
                                <1>       ;
                                <1>
                                <1>       ; pushes for subroutine stack pops compatibility
                                <1>
                                <1>       ;push  ds  ; *
                                <1>       ;push  es  ; **
                                <1>
0000390F 55                     <1>       push   ebp ; ***
00003910 56                     <1>       push   esi ; ****
                                <1>
00003911 31FF                   <1>       xor    edi, edi  ; mov edi, 0
                                <1>
00003913 803D[4E090000]03       <1>       cmp    byte [vbe3], 3
0000391A 7221                   <1>       jb     short _vbe_rmi_1
                                <1>
                                <1>       ;sub   edi, edi  ; 0 = kernel call (sign)
                                <1>       ;      ; no transfer to user's buffer
                                <1>
                                <1>       ; cx = Video mode (for 4F01h, with LFB flag)
                                <1>
0000391C 66B8014F               <1>       mov    ax, 4F01h
                                <1>
00003920 E870DFFFFF             <1>       call   _vbe3_pmfn_return_mode_info
                                <1>
00003925 6683F84F               <1>       cmp    ax, 004Fh
00003929 7533                   <1>       jne    short _vbe_rmi_2 ; fail
                                <1>
```

```
10975                                  <1>       ; 15/12/2020
10976                                  <1>       ; cx = vbe video mode
10977 0000392B 80E501                  <1>       and   ch, 01h ; clear LFB flag
10978 0000392E BEFE7B0900              <1>       mov   esi, VBE3MODEINFOBLOCK - 2
10979 00003933 66890E                  <1>       mov   [esi], cx ; MODEINFO.mode
10980 00003936 E8A6000000              <1>       call  set_lfbinfo_table
10981 0000393B EB22                    <1>       jmp   short _vbe_rmi_3  ; cf = 0
10982                                  <1> _vbe_rmi_1:
10983 0000393D 803D[4E090000]02        <1>       cmp   byte [vbe3], 2
10984 00003944 7219                    <1>       jb    short _vbe_rmi_3 ; cf = 1
10985 00003946 A0[4F090000]            <1>       mov   al, [vbe2bios] ; 0C0h-0C5h for emu (*)
10986 0000394B 3CC0                    <1>       cmp   al, 0C0h ; BOCHS/QEMU/VIRTUALBOX (*) ?
10987 0000394D 7210                    <1>       jb    short _vbe_rmi_3  ; cf = 1
10988 0000394F 3CC5                    <1>       cmp   al, 0C5h ; (*)
10989 00003951 770B                    <1>       ja    short _vbe_rmi_2 ; unknown vbios !?
10990                                  <1>
10991                                  <1>       ;xor   edi, edi  ; 0 = kernel call (sign)
10992                                  <1>       ;      ; no transfer to user's buffer
10993                                  <1>
10994                                  <1>       ;mov   ax, 4F01h
10995                                  <1>
10996                                  <1>       ; cx = Video mode (for 4F01h, with LFB flag)
10997                                  <1>
10998 00003953 E80A000000              <1>       call  vbe_biosfn_return_mode_info
10999 00003958 6683F84F                <1>       cmp   ax, 004Fh ; successful ?
11000 0000395C 7401                    <1>       je    short _vbe_rmi_3  ; cf = 0
11001                                  <1> _vbe_rmi_2:
11002 0000395E F9                      <1>       stc
11003                                  <1>       ; cf = 1
11004                                  <1> _vbe_rmi_3:
11005 0000395F 5E                      <1>       pop   esi ; ****
11006 00003960 5D                      <1>       pop   ebp ; ***
11007                                  <1>
11008                                  <1>       ;pop   es  ; **
11009                                  <1>       ;pop   ss  ; *
11010                                  <1>
11011 00003961 C3                      <1>       retn
11012                                  <1>
11013                                  <1>
11014                                  <1> ; * (TRDOS 386, INT 31h, VESA Video Bios functions)
11015                                  <1> ; * ------------------------------------------------------
11016                                  <1> ; * Function 01h - Return VBE Mode Information
11017                                  <1> ; * ------------------------------------------------------
11018                                  <1> ; * Input:
11019                                  <1> ; *         AX = 4F01h
11020                                  <1> ; *         CX = Mode number
11021                                  <1> ; *    (ES:DI) EDI = Pointer to ModeInfoBlock structure
11022                                  <1> ; * Output:
11023                                  <1> ; *         AX = VBE Return Status
11024                                  <1> ; *
11025                                  <1> ; *------------------------------------------------------
11026                                  <1> ; *
11027                                  <1>
11028                                  <1> vbe_biosfn_return_mode_info:
11029                                  <1>       ; 15/12/2020
11030                                  <1>       ; 14/12/2020
11031                                  <1>       ; 12/12/2020
11032                                  <1>       ; 11/12/2020 (TRDOS 386 v2.0.3)
11033                                  <1>       ;
11034                                  <1>       ; Input:
11035                                  <1>       ;      cx = video (bios) mode
11036                                  <1>       ;      edi = ModeInfoBlock buffer address
11037                                  <1>       ;           (in user's memory space)
11038                                  <1>       ;      (ax = 4F01h)
11039                                  <1>       ; Output:
11040                                  <1>       ;      ax = 004Fh (successful)
11041                                  <1>       ;      ah > 0 -> error
11042                                  <1>       ;
11043                                  <1>       ; Modified registers: esi
11044                                  <1>
11045                                  <1>       ;;push ds  ; *
11046                                  <1>       ;;push es  ; **
11047                                  <1>       ;;push ebp ; ***
11048                                  <1>       ;;push esi ; ****
11049                                  <1>
11050 00003962 F6C501                  <1>       test  ch, 1
11051 00003965 7505                    <1>       jnz   short vbe_rmi_1
11052                                  <1>
11053                                  <1>       ; mode number < 100h
11054                                  <1>       ; CGA/VGA mode is not proper this VBE function
11055                                  <1>
11056 00003967 29C0                    <1>       sub   eax, eax
11057                                  <1> vbe_rmi_0:
11058                                  <1>       ;mov   ax, 0100h  ; Function is not supported
11059 00003969 B401                    <1>       mov   ah, 1
11060 0000396B C3                      <1>       retn
11061                                  <1> vbe_rmi_1:
11062 0000396C 52                      <1>       push  edx ; *****
11063 0000396D 51                      <1>       push  ecx ; ******
11064 0000396E 53                      <1>       push  ebx ; *******
11065 0000396F 57                      <1>       push  edi ; ********
11066                                  <1>
11067                                  <1>       ; 14/12/2020
11068 00003970 89CB                    <1>       mov   ebx, ecx
11069                                  <1>
11070                                  <1>       ;xor   eax, eax
11071 00003972 80E7C1                  <1>       and   bh, 0C1h ; use bit 15, 14, 8 only (for bh)
11072 00003975 883D[05120300]          <1>       mov   [vbe_mode_x], bh
11073                                  <1>       ;and   bx, 1FFh
11074 0000397B 80E701                  <1>       and   bh, 1
11075                                  <1>       ;mov   bh, 1
11076                                  <1>
11077                                  <1>       ; Alternative 2 (instead of 'Mode_info_find_mode')
11078 0000397E E88A060000              <1>       call  set_mode_info_list ; (alternative 2)
11079                                  <1>
```

```
11080                                  <1>       ; eax = 0
11081                                  <1>
11082                                  <1>       ;mov   bx, [esi] ; mode
11083                                  <1>
11084                                  <1>       ; Alternative 1 (instead of 'set_mode_info_list')
11085                                  <1>       ;call  mode_info_find_mode ; (alternative 1)
11086                                  <1>
11087 00003983 09F6                    <1>       or     esi, esi
11088                                  <1>       ; 14/12/2020
11089 00003985 744D                    <1>       jz     short vbe_rmi_4     ; VBE mode number is wrong
11090                                  <1>                                  ; or it is not supported
11091                                  <1>
11092                                  <1>       ; 15/12/2020
11093                                  <1>       ;mov   bx, [esi] ; mode
11094                                  <1>
11095                                  <1>       ; 12/12/2020
11096                                  <1>       ;call  set_lfbinfo_table
11097                                  <1>
11098 00003987 F605[05120300]40        <1>       test   byte [vbe_mode_x], 40h ; LFB model ?
11099 0000398E 7404                    <1>       jz     short vbe_rmi_2
11100                                  <1>
11101 00003990 C6461C01                <1>       mov    byte [esi+MODEINFO.NumberOfBanks], 1
11102                                  <1> vbe_rmi_2:
11103                                  <1>       ; (vbe.c, 02/01/2020, vruppert)
11104                                  <1>       ; 11/12/2020 (Erdogan Tan, video.s)
11105                                  <1>       ; Bochs Graphics Adapter
11106                                  <1>       ; vendor_id: 1111h, device id: 1234h
11107                                  <1>
11108 00003994 E855070000             <1>       call   pci_get_lfb_addr
11109                                  <1>       ;or    eax, eax
11110 00003999 7404                    <1>       jz     short vbe_rmi_3
11111                                  <1>       ; zf = 0, ax > 0 (high word of LFB address)
11112                                  <1>       ; set/change LFB address in MODEINFO structure
11113 0000399B 6689462C               <1>       mov    [esi+MODEINFO.PhysBasePtr+2], ax
11114                                  <1>       ; 12/12/2020
11115                                  <1>       ;mov   [edi+LFBINFO.LFB_addr+2], ax
11116                                  <1> vbe_rmi_3:
11117                                  <1>       ;test  byte [esi+MODEINFO.WinAAttributes], 1
11118                                  <1>       ;            ; VBE_WINDOW_ATTRIBUTE_RELOCATABLE = 1
11119                                  <1>         ;jz  short vbe_rmi_4
11120                                  <1>       ;; 11/12/2020
11121                                  <1>       ;; In fact, this is far call address in (Bochs/BGA) Video Bios
11122                                  <1>       ;; Direct user access to kernel subroutines is not possible
11123                                  <1>       ;; in TRDOS 386. Also, TRDOS 386 kernel will support only LFB.
11124                                  <1>       ;; Bank select may be a seperate sysvideo function in future
11125                                  <1>       ;; (if it will be required).
11126                                  <1>       ;mov   dword [esi+MODEINFO.WinFuncPtr], dispi_set_bank_farcall
11127                                  <1> ;vbe_rmi_4:
11128                                  <1>       ; 12/12/2020
11129 0000399F E83D000000             <1>       call   set_lfbinfo_table
11130                                  <1>
11131                                  <1>       ; 11/12/2020
11132                                  <1>       ; copy 68 bytes of MODE_INFO_LIST to user
11133                                  <1>
11134 000039A4 8B3C24                 <1>       mov    edi, [esp]  ; user's buffer address
11135                                  <1>       ; 12/12/2020
11136 000039A7 09FF                    <1>       or     edi, edi ; 0 = kernel call
11137                                  <1>                       ; (call from '_vbe_biosfn_return_mode_info')
11138 000039A9 7432                    <1>       jz     short vbe_rmi_6
11139                                  <1>
11140                                  <1>       ; 15/12/2020
11141                                  <1>       ; prepare 256 bytes MODEINFO buffer at VBE3MODEINFOBLOCK
11142                                  <1>       ; and then, copy buffer conttent to user's buffer
11143 000039AB 57                     <1>       push   edi
11144 000039AC BE[24120300]            <1>       mov    esi, MODE_INFO_LIST + 2    ; MODEINFO.ModeAttributes
11145 000039B1 BF007C0900              <1>       mov    edi, VBE3MODEINFOBLOCK
11146 000039B6 B910000000              <1>       mov    ecx, 66/4  ; 66 bytes
11147 000039BB F3A5                    <1>       rep    movsd
11148 000039BD 31C0                    <1>       xor    eax, eax
11149 000039BF B12F                    <1>       mov    cl, (256-68)/4 ; 188 bytes
11150 000039C1 F3AB                    <1>       rep    stosd
11151 000039C3 66AB                    <1>       stosw  ; 2 bytes
11152 000039C5 5F                      <1>       pop    edi
11153 000039C6 BE007C0900              <1>       mov    esi, VBE3MODEINFOBLOCK
11154                                  <1>       ;mov   cx, 256
11155 000039CB FEC5                    <1>       inc    ch ; cx = 256
11156 000039CD E870DB0000              <1>       call   transfer_to_user_buffer
11157 000039D2 7309                    <1>       jnc    short vbe_rmi_5
11158                                  <1> vbe_rmi_4:
11159                                  <1>       ;mov   eax, 014Fh ; fail/error
11160 000039D4 31C0                    <1>       xor    eax, eax
11161 000039D6 B401                    <1>       mov    ah, 01h
11162                                  <1>       ;jmp   short vbe_rmi_6
11163 000039D8 E981000000              <1>       jmp    vbe_sm_ret1 ; 11/12/2020
11164                                  <1> vbe_rmi_5:
11165                                  <1>       ; 256 bytes of MODEINFO have been transferred to user
11166                                  <1>       ;mov   eax, 4Fh ; succesfull
11167                                  <1> vbe_rmi_6: ; 12/12/2020
11168 000039DD 31C0                    <1>       xor    eax, eax
11169                                  <1> ;vbe_rmi_6:
11170 000039DF EB7D                    <1>       jmp    vbe_sm_ret1 ; 11/12/2020
11171                                  <1>
11172                                  <1>       ;pop   edi ; ********
11173                                  <1>       ;pop   ebx ; *******
11174                                  <1>       ;pop   ecx ; ******
11175                                  <1>       ;pop   edx ; *****
11176                                  <1>
11177                                  <1>       ;;pop  esi ; ****
11178                                  <1>       ;;pop  ebp ; ***
11179                                  <1>       ;;pop  es  ; **
11180                                  <1>       ;;pop  ds  ; *
11181                                  <1>
11182                                  <1>       ;retn
11183                                  <1>
11184                                  <1> set_lfbinfo_table:
```

```
11185                                  <1>       ; 19/12/2020
11186                                  <1>       ; 11/12/2020
11187                                  <1>       ; Set/Fill LFBINFO structure/table
11188                                  <1>       ;
11189                                  <1>       ; Input:
11190                                  <1>       ;      esi = Mode info list address
11191                                  <1>       ; Output:
11192                                  <1>       ;      LFB_Info address is filled with LFBINFO
11193                                  <1>       ;      edi = LFB_Info address
11194                                  <1>       ;
11195                                  <1>       ; Modified registers: eax, edx (=0), edi
11196                                  <1>
11197 000039E1 BF[12120300]           <1>       mov    edi, LFB_Info
11198 000039E6 8B462A                 <1>       mov    eax, [esi+MODEINFO.PhysBasePtr]
11199 000039E9 894702                 <1>       mov    [edi+LFBINFO.LFB_addr], eax ; LFB address
11200                                  <1>       ;mov   ax, [esi+MODEINFO.mode]
11201 000039EC 668B06                 <1>       mov    ax, [esi]
11202 000039EF 668907                 <1>       mov    [edi+LFBINFO.mode],ax
11203 000039F2 8A461B                 <1>       mov    al, [esi+MODEINFO.BitsPerPixel]
11204 000039F5 88470E                 <1>       mov    [edi+LFBINFO.bpp], al
11205 000039F8 29C0                   <1>       sub    eax, eax
11206 000039FA 668B4614               <1>       mov    ax, [esi+MODEINFO.XResolution]
11207 000039FE 6689470A               <1>       mov    [edi+LFBINFO.X_res], ax
11208 00003A02 89C2                   <1>       mov    edx, eax ; 19/12/2020
11209 00003A04 668B4616               <1>       mov    ax, [esi+MODEINFO.YResolution]
11210 00003A08 6689470C               <1>       mov    [edi+LFBINFO.Y_res], ax
11211                                  <1>       ; eax = Y_res ; screen height
11212                                  <1>       ; 19/12/2020
11213 00003A0C F7E2                   <1>       mul    edx ; X_res*Y_res
11214                                  <1>       ; edx = 0
11215 00003A0E 8A570E                 <1>       mov    dl, [edi+LFBINFO.bpp]
11216                                  <1>       ; Note:
11217                                  <1>       ; Bits per pixel may be 8,16,24,32 for TRDOS 386 v2.
11218                                  <1>       ; (4 bits for pixel is not used for VESA modes here)
11219 00003A11 C0EA03                 <1>       shr    dl, 3 ; convert bits to byte
11220 00003A14 F7E2                   <1>       mul    edx
11221                                  <1>       ; eax = screen/page/buffer size in bytes
11222 00003A16 894706                 <1>       mov    [edi+LFBINFO.LFB_size], eax
11223                                  <1>       ; edx = 0
11224                                  <1>       ; clear reserved byte in LFBINFO structure/table
11225 00003A19 88570F                 <1>       mov    [edi+LFBINFO.reserved], dl ; not necessary
11226 00003A1C C3                     <1>       retn
11227                                  <1>
11228                                  <1> ; * (TRDOS 386, INT 31h, VESA Video Bios functions)
11229                                  <1> ; * ------------------------------------------------------------
11230                                  <1> ; * Function 02h - Set VBE Mode
11231                                  <1> ; * ------------------------------------------------------------
11232                                  <1> ; * Input:
11233                                  <1> ; *          AX = 4F02h
11234                                  <1> ; *          BX = Desired Mode to set
11235                                  <1> ; * Output:
11236                                  <1> ; *          AX = VBE Return Status
11237                                  <1> ; *
11238                                  <1> ; *------------------------------------------------------------
11239                                  <1> ; *
11240                                  <1>
11241                                  <1> vbe_biosfn_set_mode:
11242                                  <1>       ; 07/03/2021
11243                                  <1>       ; 12/12/2020
11244                                  <1>       ; 11/12/2020 (LFBINFO table for VESA VBE modes)
11245                                  <1>       ; 27/11/2020
11246                                  <1>       ; 25/11/2020
11247                                  <1>       ; 23/11/2020 (TRDOS 386 v2.0.3)
11248                                  <1>       ; (ref: vbe.c, 02/01/2020, vruppert)
11249                                  <1>       ;
11250                                  <1>       ; Input:
11251                                  <1>       ;      bx = video (bios) mode
11252                                  <1>       ;      ax = 4F02h
11253                                  <1>       ; Output:
11254                                  <1>       ;      ax = 004Fh (successful)
11255                                  <1>       ;      ah > 0 -> error
11256                                  <1>       ;
11257                                  <1>       ; Modified registers: esi
11258                                  <1>
11259                                  <1>       ; 27/11/2020
11260                                  <1>
11261                                  <1>       ;;push ds  ; *
11262                                  <1>       ;;push es  ; **
11263                                  <1>       ;;push ebp ; ***
11264                                  <1>       ;;push esi ; ****
11265                                  <1>
11266                                  <1>       ; 11/12/2020
11267 00003A1D 52                     <1>       push   edx ; *****
11268 00003A1E 51                     <1>       push   ecx ; ******
11269 00003A1F 53                     <1>       push   ebx ; *******
11270 00003A20 57                     <1>       push   edi ; ********
11271                                  <1>
11272                                  <1>       ;xor   eax, eax
11273 00003A21 80E7C1                 <1>       and    bh, 0C1h ; use bit 15, 14, 8 only (for bh)
11274 00003A24 883D[05120300]         <1>       mov    [vbe_mode_x], bh
11275 00003A2A 80E701                 <1>       and    bh, 1
11276 00003A2D 753C                   <1>       jnz    short vbe_sm_3  ; VESA VBE mode
11277                                  <1>
11278                                  <1>       ;;test bx, 4000h ; VBE_MODE_LINEAR_FRAME_BUFFER
11279                                  <1>       ;test  bh, 40h
11280                                  <1>       ;jz    short vbe_sm_0
11281                                  <1>       ;; lfb_flag
11282                                  <1>       ;mov   al, 40h      ; VBE_DISPI_LFB_ENABLED
11283                                  <1> vbe_sm_0:
11284                                  <1>       ; 27/11/2020
11285 00003A2F B080                   <1>       mov    al, 80h
11286                                  <1>       ;test  bh, 80h ; VBE_MODE_PRESERVE_DISPLAY_MEMORY
11287                                  <1>       ;jnz   short vbe_sm_1 ; no_clear
11288                                  <1>       ;; clear
11289                                  <1>       ;sub   al, al ; 0
```

```
11290 00003A31 8405[05120300]      <1>       test   [vbe_mode_x], al ; 80h
11291 00003A37 7402                <1>       jz     short vbe_sm_1 ; clear display memory
11292                              <1>       ; no_clear
11293 00003A39 08C3                <1>       or     bl, al ; VBE_MODE_PRESERVE_DISPLAY_MEMORY
11294                              <1> vbe_sm_1:
11295                              <1>       ; check non vesa mode
11296                              <1>       ;;cmp  bx, 100h ; VBE_MODE_VESA_DEFINED
11297                              <1>       ;;jna  short vbe_sm_2
11298                              <1>       ;and   bh, 1
11299                              <1>       ;jnz   short vbe_sm_3
11300                              <1>
11301                              <1>       ; BX <= 1FFh
11302                              <1>
11303                              <1>       ; 27/11/2020
11304                              <1>       ;or    bl, al ; al = 80h if no_clear option is set
11305                              <1>       ;             ; al = 0 if no_clear option is not set
11306                              <1>
11307                              <1>       ; 25/11/2020
11308                              <1>       ; VBE DISPI will be disabled in 'biosfn_set_video_mode'
11309                              <1>
11310                              <1>       ;xor   al, al ; 0 ; VBE_DISPI_DISABLED
11311                              <1>       ;call  dispi_set_enable
11312                              <1>
11313                              <1>       ; call the vgabios in order to set the video mode
11314                              <1>       ; this allows for going back to textmode with a VBE call
11315                              <1>       ; (some applications expect that to work)
11316                              <1>
11317                              <1>       ;and   bx, 0FFh
11318                              <1>
11319                              <1>       ; 27/11/2020
11320                              <1> biosfn_set_video_mode:
11321                              <1>       ; _call: call subroutine
11322                              <1>       ; 26/11/2020 (TRDOS 386 v2.0.3)
11323                              <1>       ; (ref: vgabios.c, 02/01/2020, vruppert)
11324                              <1>       ; Input:
11325                              <1>       ;      bl = VGA video (bios) mode
11326                              <1>       ; Output:
11327                              <1>       ;      cf = 1 -> error
11328                              <1>       ;      cf = 0 -> ok
11329                              <1>       ;
11330                              <1>       ; Modified registers: esi
11331                              <1>
11332                              <1>       ; 'dispi_set_enable(VBE_DISPI_DISABLED);'
11333                              <1>
11334                              <1>       ;mov   ax, 0 ; VBE_DISPI_DISABLED
11335 00003A3B 31C0               <1>       xor    eax, eax ; 0
11336 00003A3D E8A3040000         <1>       call   dispi_set_enable
11337                              <1>
11338 00003A42 88D8               <1>       mov    al, bl
11339                              <1>       ;jmp   _set_mode ; (in 'biosfn_set_video_mode' sub)
11340 00003A44 E8B1E0FFFF         <1>       call   _set_mode ; will return with cf=1 only if
11341                              <1>                        ; desired mode is not implemented
11342                              <1>       ; _retn: return from subroutine
11343 00003A49 721A               <1>       jc     short vbe_sm_2 ; 25/11/2020
11344                              <1>
11345                              <1>       ; 26/11/2020
11346 00003A4B 31C0               <1>       xor    eax, eax
11347 00003A4D A0[BA6A0000]       <1>       mov    al, [CRT_MODE]
11348                              <1>       ; 27/11/2020
11349 00003A52 8A25[3F8D0100]     <1>       mov    ah, [noclearmem] ; 80h or 0
11350                              <1>       ;and   ah 80h
11351 00003A58 66A3[06120300]     <1>       mov    [video_mode], ax ; bit 15 = no_clear flag
11352                              <1>                               ; bit 14 = 0 (not LFB model)
11353                              <1> vbe_sm_ret1:
11354                              <1>       ; 11/12/2020
11355                              <1>       ; (vbe_rmi_4 and vbe_rmi_6 jump here)
11356                              <1>       ; 27/11/2020
11357 00003A5E B04F               <1>       mov    al, 4Fh ; Function call successful
11358                              <1>       ; eax = 004Fh
11359                              <1> vbe_sm_ret2:
11360                              <1>       ; 11/12/2020
11361 00003A60 5F                 <1>       pop    edi ; ********
11362 00003A61 5B                 <1>       pop    ebx ; *******
11363 00003A62 59                 <1>       pop    ecx ; ******
11364 00003A63 5A                 <1>       pop    edx ; *****
11365                              <1>
11366                              <1>       ;;pop  esi ; ****
11367                              <1>       ;;pop  ebp ; ***
11368                              <1>       ;;pop  es  ; **
11369                              <1>       ;;pop  ds  ; *
11370                              <1>
11371 00003A64 C3                 <1>       retn
11372                              <1>
11373                              <1> vbe_sm_2:
11374                              <1>       ;mov   ax, 0100h ; Function is not supported
11375                              <1>       ; 27/11/2020
11376 00003A65 31C0               <1>       xor    eax, eax
11377 00003A67 B401               <1>       mov    ah, 01h
11378                              <1>       ; eax = 0100h
11379                              <1>       ;retn
11380 00003A69 EBF5               <1>       jmp    short vbe_sm_ret2
11381                              <1>
11382                              <1> vbe_sm_3:
11383                              <1>       ; 12/12/2020
11384                              <1>       ; check current mode, if it is 03h
11385                              <1>       ; save page contents and cursor positions
11386 00003A6B 803D[BA6A0000]03   <1>       cmp    byte [CRT_MODE], 03h
11387                              <1>       ;jne   short vbe_sm_0
11388 00003A72 7505               <1>       jne    short vbe_sm_4 ; 07/03/2021
11389 00003A74 E822E3FFFF         <1>       call   save_mode3_multiscreen
11390                              <1>       ; set current mode to extended (SVGA) mode
11391                              <1>       ;mov   byte [CRT_MODE], 0FFh ; VESA VBE mode
11392                              <1> vbe_sm_4:
11393                              <1>       ; 27/11/2020
11394                              <1>       ; bx = mode (bit 0 to 8)
```

```
11395                                  <1>
11396                                  <1>       ; 25/11/2020
11397                                  <1>
11398                                  <1>       ; Alternative 2 (instead of 'Mode_info_find_mode')
11399                                  <1>       ;push  edi
11400 00003A79 E88F050000               <1>       call   set_mode_info_list ; (alternative 2)
11401                                  <1>       ;pop   edi
11402                                  <1>
11403                                  <1>       ;mov   bx, [esi] ; mode
11404                                  <1>
11405                                  <1>       ; Alternative 1 (instead of 'set_mode_info_list')
11406                                  <1>       ;call  mode_info_find_mode ; (alternative 1)
11407                                  <1>
11408 00003A7E 09F6                     <1>       or     esi, esi
11409 00003A80 74E3                     <1>       jz     short vbe_sm_2 ; VBE mode number is wrong
11410                                  <1>                             ; or it is not supported
11411                                  <1>
11412                                  <1>       ; 11/12/2020
11413 00003A82 668B1E                   <1>       mov    bx, [esi] ; mode
11414                                  <1>
11415                                  <1>       ; 27/11/2020
11416 00003A85 0A3D[05120300]           <1>       or     bh, [vbe_mode_x]
11417                                  <1>
11418                                  <1>       ; save VESA VBE mode
11419 00003A8B 66891D[06120300]         <1>       mov    [video_mode], bx
11420                                  <1>              ; 27/11/2020
11421                                  <1>              ; bit 0 to 8 = VESA VBE mode
11422                                  <1>              ; bit 9 to 13 = 0 (bit 0 to 13 = mode)
11423                                  <1>              ; bit 14 = Linear/Flat Frame Buffer flag
11424                                  <1>              ; bit 15 = 'memory not cleared
11425                                  <1>              ;         at last mode set' flag
11426                                  <1>
11427                                  <1>       ; first disable current mode
11428                                  <1>       ; (when switching between vesa modes)
11429                                  <1>       ; 'dispi_set_enable(VBE_DISPI_DISABLED);'
11430                                  <1>
11431                                  <1>       ;mov   ax, VBE_DISPI_DISABLED ; 0
11432 00003A92 29C0                     <1>       sub    eax, eax ; 0
11433                                  <1>
11434 00003A94 E84C040000               <1>       call   dispi_set_enable
11435                                  <1>
11436                                  <1>       ; 11/12/2020
11437 00003A99 8A461B                   <1>       mov    al, [esi+MODEINFO.BitsPerPixel]
11438                                  <1>       ; ah = 0
11439                                  <1>
11440                                  <1>       ;cmp   byte [esi+MODEINFO.BitsPerPixel], 8
11441 00003A9C 3C08                     <1>       cmp    al, 8
11442 00003A9E 750B                     <1>       jne    short vbe_sm_5
11443                                  <1>
11444                                  <1>       ; 11/12/2020
11445                                  <1>       ;push  edi
11446 00003AA0 50                       <1>       push   eax
11447                                  <1>       ; 'load_dac_palette(3);'
11448 00003AA1 56                       <1>       push   esi
11449 00003AA2 B403                     <1>       mov    ah, 3  ; palette3, 256 colors
11450 00003AA4 E83AF1FFFF               <1>       call   load_dac_palette
11451 00003AA9 5E                       <1>       pop    esi
11452                                  <1>       ; 11/12/2020
11453 00003AAA 58                       <1>       pop    eax
11454                                  <1>       ;pop   edi
11455                                  <1> vbe_sm_5:
11456                                  <1>       ;'dispi_set_bpp(cur_info->info.BitsPerPixel);'
11457                                  <1>       ; 11/12/2020 (al = bits per pixel, ah = 0)
11458                                  <1>       ;xor   ah, ah
11459                                  <1>       ;mov   al, [esi+MODEINFO.BitsPerPixel]
11460 00003AAB E849040000               <1>       call   dispi_set_bpp
11461                                  <1>         ;'dispi_set_xres(cur_info->info.XResolution);'
11462 00003AB0 668B4614                 <1>       mov    ax, [esi+MODEINFO.XResolution]
11463 00003AB4 E846040000               <1>       call   dispi_set_xres
11464                                  <1>         ;'dispi_set_yres(cur_info->info.YResolution);'
11465 00003AB9 668B4616                 <1>       mov    ax, [esi+MODEINFO.YResolution]
11466 00003ABD E843040000               <1>       call   dispi_set_yres
11467                                  <1>
11468                                  <1>       ;'dispi_set_bank(0);'
11469                                  <1>       ;xor   ax, ax
11470 00003AC2 31C0                     <1>       xor    eax, eax ; 0
11471 00003AC4 E842040000               <1>       call   dispi_set_bank
11472                                  <1>         ;'dispi_set_enable(VBE_DISPI_ENABLED|no_clear|lfb_flag);'
11473                                  <1>       ;mov   ax, di
11474                                  <1>       ; ah = 0 ; 27/11/2020
11475 00003AC9 A0[05120300]             <1>       mov    al, [vbe_mode_x] ; restore VBE mode bit 14 & 15
11476 00003ACE 0C01                     <1>       or     al, 1 ; VBE_DISPI_ENABLED
11477 00003AD0 E810040000               <1>       call   dispi_set_enable
11478                                  <1>
11479                                  <1>         ; 'vga_compat_setup();'
11480 00003AD5 E846040000               <1>       call   vga_compat_setup
11481                                  <1>
11482                                  <1>       ; 11/12/2020
11483 00003ADA E802FFFFFF               <1>       call   set_lfbinfo_table
11484                                  <1>
11485                                  <1>       ; 26/11/2020
11486 00003ADF 31C0                     <1>       xor    eax, eax
11487 00003AE1 FEC8                     <1>       dec    al
11488 00003AE3 A2[BA6A0000]             <1>       mov    [CRT_MODE], al ; 0FFh = VESA VBE mode sign
11489                                  <1>
11490                                  <1>       ; 27/11/2020
11491 00003AE8 E971FFFFFF               <1>       jmp    vbe_sm_ret1 ; Function call successful
11492                                  <1>
11493                                  <1>       ; 27/11/2020
11494                                  <1>       ;mov   al, 4Fh
11495                                  <1>       ;      ; eax = 004Fh = Function call successful
11496                                  <1>       ;jmp   short vbe_sm_ret2
11497                                  <1>
11498                                  <1> ; * (TRDOS 386, INT 31h, VESA Video Bios functions)
11499                                  <1> ; * ------------------------------------------------------
```

```
11500                                 <1> ; * Function 03h - Return Current VBE Mode
11501                                 <1> ; * ------------------------------------------------------------
11502                                 <1> ; * Input:
11503                                 <1> ; *          AX = 4F03h
11504                                 <1> ; * Output:
11505                                 <1> ; *          AX = VBE Return Status
11506                                 <1> ; *          BX = Current VBE Mode
11507                                 <1> ; *
11508                                 <1> ; *------------------------------------------------------------
11509                                 <1> ; *
11510                                 <1>
11511                                 <1> vbe_biosfn_return_current_mode:
11512                                 <1>       ; 11/12/2020
11513                                 <1>       ; 27/11/2020 (TRDOS 386 v2.0.3)
11514                                 <1>       ; (ref: vbe.c, 02/01/2020, vruppert)
11515                                 <1>       ;
11516                                 <1>       ; Input:
11517                                 <1>       ;      none
11518                                 <1>       ; Output:
11519                                 <1>       ;      ax = 004Fh (successful)
11520                                 <1>       ;      ah > 0 -> error
11521                                 <1>       ;      bx = current video (bios) mode (if ah = 0)
11522                                 <1>       ;
11523                                 <1>       ; Modified registers: eax, ebx
11524                                 <1>
11525                                 <1>       ; 27/11/2020
11526                                 <1>
11527                                 <1>       ;;push ds  ; *
11528                                 <1>       ;;push es  ; **
11529                                 <1>       ;;push ebp ; ***
11530                                 <1>       ;;push esi ; ****
11531                                 <1>
11532                                 <1>       ;push  edx ; *****
11533                                 <1>
11534                                 <1>       ; (vbe.c)
11535                                 <1>       ;call  dispi_get_enable
11536                                 <1>       ;      ; ax = vbe display interface status
11537                                 <1>       ;and   al, 1 ; VBE_DISPI_ENABLED
11538                                 <1>       ;jnz   short vbe_gm_1  ; VBE graphics mode
11539                                 <1>
11540 00003AED A0[BA6A0000]          <1>       mov    al, [CRT_MODE] ; current cga/vga mode
11541 00003AF2 3CFF                  <1>       cmp    al, 0FFh ; VBE extension signature
11542 00003AF4 720E                  <1>       jb     short vbe_gm_1 ; get CGA/VGA mode
11543                                 <1>
11544                                 <1>       ; get VBE mode
11545                                 <1> vbe_gm_0:
11546 00003AF6 66A1[06120300]        <1>       mov    ax, [video_mode]
11547                                 <1>              ; BX bits:
11548                                 <1>              ; bit 0 to 8 = VESA VBE video mode
11549                                 <1>              ; bit 9 to 13 = 0
11550                                 <1>              ; bit 14 = last mode set LFB option
11551                                 <1>              ;         1 - linear/flat frame buffer
11552                                 <1>              ;         0 - windowed frame buffer
11553                                 <1>              ; bit 15 = last mode set no_clear option
11554                                 <1>              ;         0 - video memory cleared
11555                                 <1>              ;         1 - video memory not cleared
11556                                 <1>
11557                                 <1> vbe_gm_return:
11558                                 <1>       ;pop   edx ; ******
11559 00003AFC 0FB7D8                <1>       movzx  ebx, ax
11560                                 <1> ;vbe_srs_retn:
11561 00003AFF 31C0                  <1>       xor    eax, eax ; 0
11562 00003B01 B04F                  <1>       mov    al, 4Fh ; ax = 004Fh (successful)
11563 00003B03 C3                    <1>       retn
11564                                 <1>
11565                                 <1> vbe_gm_1:
11566                                 <1>       ; legacy (old, standard) CGA/VGA bios video mode
11567 00003B04 8A25[3F8D0100]        <1>       mov    ah, [noclearmem] ; 80h or 0
11568                                 <1>              ; BX bits:
11569                                 <1>              ; bit 0 to 7 = video mode
11570                                 <1>              ; bit 8 to 13 = 0
11571                                 <1>              ; bit 14 = 0 (not LFB mode) CGA/VGA
11572                                 <1>              ; bit 15 = 1 if [noclearmem] = 80h
11573                                 <1>              ;         0 if [noclearmem] = 0
11574 00003B0A EBF0                  <1>         jmp    short vbe_gm_return
11575                                 <1>
11576                                 <1> ; * (TRDOS 386, INT 31h, VESA Video Bios functions)
11577                                 <1> ; * ------------------------------------------------------------
11578                                 <1> ; * Function 04h - Save/Restore State
11579                                 <1> ; * ------------------------------------------------------------
11580                                 <1> ; * Input:
11581                                 <1> ; *          AX = 4F04h
11582                                 <1> ; *              DL = 00h Return Save/Restore State buff size
11583                                 <1> ; *                   01h Save State
11584                                 <1> ; *                   02h Restore State
11585                                 <1> ; *              CX = Requested states
11586                                 <1> ; *                bit 0 - controller hardware state
11587                                 <1> ; *                bit 1 - BIOS data state
11588                                 <1> ; *                bit 2 - DAC state
11589                                 <1> ; *                bit 3 - register state
11590                                 <1> ; *    (ES:BX) EBX = Pointer to buffer (if DL <> 00h)
11591                                 <1> ; * Output:
11592                                 <1> ; *          AX = VBE Return Status
11593                                 <1> ; *          BX = Number of 64-byte blocks
11594                                 <1> ; *               to hold the state buffer (if DL=00h)
11595                                 <1> ; *
11596                                 <1> ; *------------------------------------------------------------
11597                                 <1> ; *
11598                                 <1>
11599                                 <1> vbe_biosfn_save_restore_state:
11600                                 <1>       ; 23/01/2021
11601                                 <1>       ; 16/01/2021
11602                                 <1>       ; 14/01/2021
11603                                 <1>       ; 13/01/2021
11604                                 <1>       ; 12/01/2021
```

```
11605                                  <1>       ; 11/01/2021 (TRDOS 386 v2.0.3)
11606                                  <1>       ; (ref: vbe.c, 02/01/2020, vruppert)
11607                                  <1>       ;
11608                                  <1>       ; Input:
11609                                  <1>       ;      dl = sub function
11610                                  <1>       ;      cl = requested state
11611                                  <1>       ;       ebx = pointer to buffer (if dl<>00h)
11612                                  <1>       ; Output:
11613                                  <1>       ;      ax = 004Fh (successful)
11614                                  <1>       ;      ah > 0 -> error
11615                                  <1>       ;      bx = Number of 64-byte blocks
11616                                  <1>       ;           to hold the state buffer (if DL=00h)
11617                                  <1>
11618                                  <1>       ; Modified registers: eax, ebx, edi
11619                                  <1>
11620                                  <1>       ; 14/01/2021
11621 00003B0C 09DB                   <1>       or     ebx, ebx ; user's buffer address
11622 00003B0E 750A                   <1>       jnz    short _vbe_biosfn_save_restore_state
11623                                  <1>
11624 00003B10 20D2                   <1>       and    dl, dl
11625 00003B12 7406                   <1>       jz     short _vbe_biosfn_save_restore_state
11626                                  <1>
11627                                  <1>       ; function failed
11628                                  <1>       ;mov   eax, 0100h
11629                                  <1>       ;xor   eax, eax
11630                                  <1>       ;inc   ah  ; eax = 0100h
11631                                  <1>       ; 16/01/2021
11632 00003B14 B84F010000             <1>       mov    eax, 014Fh
11633 00003B19 C3                     <1>       retn
11634                                  <1>
11635                                  <1> _vbe_biosfn_save_restore_state:
11636                                  <1>       ; 23/01/2021
11637                                  <1>       ; 14/01/2021
11638                                  <1>       ; ebx = 0 if the caller is kernel ('sysvideo')
11639                                  <1>
11640                                  <1>       ; 13/01/2021
11641 00003B1A 57                     <1>       push   edi
11642 00003B1B 52                     <1>       push   edx
11643 00003B1C 51                     <1>       push   ecx
11644                                  <1>
11645                                  <1>       ; 23/01/2021
11646                                  <1>       ; 12/01/2021
11647 00003B1D 80FA02                 <1>       cmp    dl, 2
11648 00003B20 7757                   <1>       ja     short vbe_srs_7 ; 23/01/2021
11649                                  <1>                      ; invalid sub function
11650 00003B22 83F90F                 <1>       cmp    ecx, 0Fh
11651 00003B25 7752                   <1>       ja     short vbe_srs_7 ; invalid !
11652                                  <1>
11653 00003B27 20D2                   <1>       and    dl, dl
11654 00003B29 7515                   <1>       jnz    short vbe_srs_4
11655                                  <1>
11656                                  <1>       ; DL = 0
11657                                  <1>       ; Return Save/Restore State buffer size
11658                                  <1>
11659                                  <1>       ;mov   ebx, ecx
11660                                  <1>       ;shl   bl, 1
11661                                  <1>       ;mov   bx, [ebx+vbestatebufsize]
11662 00003B2B E881000000             <1>       call   vbe_srs_gbs
11663                                  <1>
11664                                  <1> ;     ; 11/01/2021
11665                                  <1> ;     test   cl, 8
11666                                  <1> ;     jz     short vbe_srs_3
11667                                  <1> ;     ; vbe_biosfn_read_video_state_size();
11668                                  <1> ;     ; return 9 * 2;
11669                                  <1> ;     mov    bl, 18 ; register state size
11670                                  <1> ;vbe_srs_0:
11671                                  <1> ;     test   cl, 1
11672                                  <1> ;     jz     short vbe_srs_1
11673                                  <1> ;     ; size += 0x46;
11674                                  <1> ;     add    bl, 70 ; controller state size
11675                                  <1> ;vbe_srs_1:
11676                                  <1> ;     test   cl, 2
11677                                  <1> ;     jz     short vbe_srs_2
11678                                  <1> ;     ; size += (5 + 8 + 5) * 2 + 6;
11679                                  <1> ;     ;add   bl, 42 ; BIOS data state size ; Bochs/Plex86
11680                                  <1> ;     ; 12/01/2021
11681                                  <1> ;     add    bl, 40 ; TRDOS 386 v2 VBIOS data state size
11682                                  <1> ;vbe_srs_2:
11683                                  <1> ;     test   cl, 4
11684                                  <1> ;     jz     short vbe_srs_3
11685                                  <1> ;     ; size += 3 + 256 * 3 + 1;
11686                                  <1> ;     add    bx, 772 ; DAC state size
11687                                  <1>
11688                                  <1> vbe_srs_3:
11689 00003B30 6683C33F               <1>       add    bx, 63
11690 00003B34 66C1EB06               <1>       shr    bx, 6 ; / 64
11691                                  <1>
11692                                  <1> vbe_srs_retn:
11693 00003B38 31C0                   <1>       xor    eax, eax ; 0
11694                                  <1> vbe_srs_0:  ; 16/01/2021
11695 00003B3A B04F                   <1>       mov    al, 4Fh ; ax = 004Fh (successful)
11696                                  <1> ;vbe_srs_0:
11697                                  <1>       ; 13/01/2021
11698 00003B3C 59                     <1>       pop    ecx
11699 00003B3D 5A                     <1>       pop    edx
11700 00003B3E 5F                     <1>       pop    edi
11701                                  <1>
11702 00003B3F C3                     <1>       retn
11703                                  <1>
11704                                  <1>       ; 23/01/2021
11705                                  <1> ;vbe_srs_10:
11706                                  <1>       ;; 14/01/2021
11707                                  <1>       ; return to 'sysvideo'
11708                                  <1>       ;mov   ebx, ecx ; transfer count
11709                                  <1>       ;      ; (byte count for saving current video state)
```

```
11710                                  <1>       ;jmp   short vbe_srs_retn
11711                                  <1>
11712                                  <1> vbe_srs_4:
11713                                  <1>       ; 23/01/2021
11714 00003B40 80E10F                  <1>       and    cl, 0Fh ; 8, 4, 2, 1
11715 00003B43 7434                    <1>       jz     short vbe_srs_7 ; cx = 0 -> invalid !
11716                                  <1>
11717 00003B45 BF00760900              <1>       mov    edi, VBE3SAVERESTOREBLOCK
11718                                  <1>
11719 00003B4A 80FA01                  <1>       cmp    dl, 1
11720 00003B4D 7730                    <1>       ja     short vbe_srs_8
11721                                  <1>
11722                                  <1>       ; save video state
11723                                  <1>
11724 00003B4F F6C107                  <1>       test   cl, 07h ; 4, 2, 1
11725 00003B52 740A                    <1>       jz     short vbe_srs_5  ; vbe dispi regs state
11726                                  <1>
11727 00003B54 E884000000              <1>       call   biosfn_save_video_state
11728                                  <1>       ; edi = current position
11729                                  <1>       ;      in VBE3SAVERESTOREBLOCK
11730                                  <1>       ;      (VGA save_state offset)
11731                                  <1>       ; modified regs: edi, eax, edx, ch
11732 00003B59 F6C108                  <1>       test   cl, 8
11733 00003B5C 7405                    <1>       jz     short vbe_srs_6
11734                                  <1> vbe_srs_5:
11735 00003B5E E8AC010000              <1>       call   vbe_biosfn_save_video_state
11736                                  <1>       ; edi = end position
11737                                  <1>       ;      in VBE3SAVERESTOREBLOCK
11738                                  <1>       ;      (VGA save_state offset)
11739                                  <1>       ; modified regs: edi, eax, edx, ch
11740                                  <1> vbe_srs_6:
11741                                  <1>       ; 23/01/2021
11742 00003B63 21DB                    <1>       and    ebx, ebx
11743 00003B65 74D1                    <1>       jz     short vbe_srs_retn ; the caller is kernel
11744                                  <1>
11745 00003B67 BE00760900              <1>       mov    esi, VBE3SAVERESTOREBLOCK
11746 00003B6C 29F7                    <1>       sub    edi, esi
11747 00003B6E 89F9                    <1>       mov    ecx, edi ; transfer count in bytes
11748                                  <1>
11749                                  <1>       ;; 14/01/2021
11750                                  <1>       ;and   ebx, ebx
11751                                  <1>       ;jz    short vbe_srs_10 ; the caller is kernel
11752                                  <1>
11753 00003B70 89DF                    <1>       mov    edi, ebx ; user's buffer address
11754 00003B72 E8CBD90000              <1>       call   transfer_to_user_buffer
11755 00003B77 73BF                    <1>       jnc    short vbe_srs_retn
11756                                  <1> vbe_srs_7:
11757                                  <1>       ; // function failed
11758                                  <1>       ;mov   eax, 0100h
11759 00003B79 31C0                    <1>       xor    eax, eax
11760 00003B7B FEC4                    <1>       inc    ah  ; eax = 0100h
11761                                  <1>       ; 16/01/2021
11762                                  <1>       ; ax = 0014Fh
11763                                  <1>       ;retn
11764                                  <1>       ; 13/01/2021
11765 00003B7D EBBB                    <1>       jmp    short vbe_srs_0
11766                                  <1> vbe_srs_8:
11767                                  <1>       ;cmp   dl, 2
11768                                  <1>       ;jne   short vbe_srs_7
11769                                  <1>       ;            ; invalid sub function
11770                                  <1>
11771                                  <1>       ; 14/01/2021
11772 00003B7F 09DB                    <1>       or     ebx, ebx ; user's buffer address
11773                                  <1>       ;jnz   short vbe_srs_11
11774                                  <1>
11775                                  <1>       ; the caller is kernel ('sysvideo')
11776                                  <1>       ;jmp   short vbe_srs_12
11777                                  <1>       ; 23/01/2021
11778 00003B81 7414                    <1>       jz     short vbe_srs_12 ; 'sysvideo' call
11779                                  <1> vbe_srs_11:
11780 00003B83 89DE                    <1>       mov    esi, ebx ; user's buffer address
11781                                  <1>       ; 23/01/2021
11782                                  <1>       ;push  ebx
11783                                  <1>
11784 00003B85 E827000000              <1>       call   vbe_srs_gbs
11785                                  <1>
11786                                  <1>       ; restore video state
11787                                  <1>
11788                                  <1>       ;mov   edi, VBE3SAVERESTOREBLOCK
11789 00003B8A 51                      <1>       push   ecx
11790 00003B8B 89D9                    <1>       mov    ecx, ebx ; transfer count in bytes
11791 00003B8D E8FAD90000              <1>       call   transfer_from_user_buffer
11792 00003B92 59                      <1>       pop    ecx
11793                                  <1>       ; 23/01/2021
11794                                  <1>       ;pop   ebx
11795 00003B93 89F3                    <1>       mov    ebx, esi
11796 00003B95 72E2                    <1>       jc     short vbe_srs_7
11797                                  <1>
11798                                  <1> vbe_srs_12:
11799                                  <1>       ;mov   esi, VBE3SAVERESTOREBLOCK
11800 00003B97 89FE                    <1>       mov    esi, edi
11801                                  <1>
11802 00003B99 F6C107                  <1>       test   cl, 07h ; 4, 2, 1
11803 00003B9C 740C                    <1>       jz     short vbe_srs_9     ; vbe dispi regs state
11804                                  <1>
11805 00003B9E E8A8010000              <1>       call   biosfn_restore_video_state
11806 00003BA3 72D4                    <1>       jc     short vbe_srs_7 ; invalid buffer content !
11807                                  <1>       ; esi = current position
11808                                  <1>       ;      in VBE3SAVERESTOREBLOCK
11809                                  <1>       ;      (VGA save_state offset)
11810                                  <1>       ; modified regs: esi, eax, edx, ch
11811 00003BA5 F6C108                  <1>       test   cl, 8
11812                                  <1>       ;jz    short vbe_srs_10
11813                                  <1>       ; 23/01/2020
11814 00003BA8 EB8E                    <1>       jmp    short vbe_srs_retn
```

```
11815                                  <1> vbe_srs_9:
11816 00003BAA E8F8020000              <1>       call   vbe_biosfn_restore_video_state
11817                                  <1>
11818                                  <1>       ; modified regs: esi, eax, edx, ch
11819                                  <1>
11820 00003BAF EB87                    <1>       jmp    short vbe_srs_retn
11821                                  <1>
11822                                  <1> ;vbe_srs_10:
11823                                  <1> ;     ; successful
11824                                  <1> ;     xor    eax, eax ; 0
11825                                  <1> ;     mov    al, 4Fh ; ax = 004Fh (successful)
11826                                  <1> ;     retn
11827                                  <1>
11828                                  <1> vbe_srs_gbs:
11829                                  <1>       ; return buffer size according to flags
11830 00003BB1 89CB                    <1>       mov    ebx, ecx ; options/flags
11831 00003BB3 D0E3                    <1>       shl    bl, 1
11832 00003BB5 668B9B[BD3B0000]        <1>       mov    bx, [ebx+vbestatebufsize]
11833 00003BBC C3                      <1>       retn
11834                                  <1>
11835                                  <1> vbestatebufsize:
11836                                  <1>       ; ---------------------------------------
11837                                  <1>       ; CL = 0  1   2    3    4    5    6    7
11838                                  <1>       ; ---------------------------------------
11839 00003BBD 0000460028006E0004-     <1>       dw     0, 70, 40, 110, 772, 842, 812, 882
11839 00003BC6 034A032C037203          <1>
11840                                  <1>       ; ---------------------------------------
11841                                  <1>       ; CL = 8   9   10  11   12   13   14   15
11842                                  <1>       ; ---------------------------------------
11843 00003BCD 120058003A00800016-     <1>       dw     18, 88, 58, 128, 790, 860, 830, 900
11843 00003BD6 035C033E038403          <1>
11844                                  <1>
11845                                  <1> ; 11/01/2021
11846                                  <1> VGAREG_ACTL_ADDRESS       equ 3C0h
11847                                  <1> VGAREG_ACTL_WRITE_DATA    equ 3C0h
11848                                  <1> VGAREG_ACTL_READ_DATA     equ 3C1h
11849                                  <1>
11850                                  <1> VGAREG_INPUT_STATUS       equ 3C2h
11851                                  <1> VGAREG_WRITE_MISC_OUTPUT equ 3C2h
11852                                  <1> VGAREG_VIDEO_ENABLE       equ 3C3h
11853                                  <1> VGAREG_SEQU_ADDRESS       equ 3C4h
11854                                  <1> VGAREG_SEQU_DATA   equ 3C5h
11855                                  <1>
11856                                  <1> VGAREG_PEL_MASK           equ 3C6h
11857                                  <1> VGAREG_DAC_STATE   equ 3C7h
11858                                  <1> VGAREG_DAC_READ_ADDRESS   equ 3C7h
11859                                  <1> VGAREG_DAC_WRITE_ADDRESS equ 3C8h
11860                                  <1> VGAREG_DAC_DATA           equ 3C9h
11861                                  <1>
11862                                  <1> VGAREG_READ_FEATURE_CTL   equ 3CAh
11863                                  <1> VGAREG_READ_MISC_OUTPUT   equ 3CCh
11864                                  <1>
11865                                  <1> VGAREG_GRDC_ADDRESS       equ 3CEh
11866                                  <1> VGAREG_GRDC_DATA   equ 3CFh
11867                                  <1>
11868                                  <1> ;VGAREG_MDA_CRTC_ADDRESS equ 3B4h
11869                                  <1> ;VGAREG_MDA_CRTC_DATA     equ 3B5h
11870                                  <1> VGAREG_VGA_CRTC_ADDRESS   equ 3D4h
11871                                  <1> VGAREG_VGA_CRTC_DATA      equ 3D5h
11872                                  <1>
11873                                  <1> ;VGAREG_MDA_WRITE_FEATURE_CTL equ 3BAh
11874                                  <1> VGAREG_VGA_WRITE_FEATURE_CTL equ 3DAh
11875                                  <1> VGAREG_ACTL_RESET  equ 3DAh
11876                                  <1>
11877                                  <1> ;VGAREG_MDA_MODECTL       equ 3B8h
11878                                  <1> VGAREG_CGA_MODECTL equ 3D8h
11879                                  <1> VGAREG_CGA_PALETTE equ 3D9h
11880                                  <1>
11881                                  <1> biosfn_save_video_state:
11882                                  <1>       ; 22/01/2021
11883                                  <1>       ; 12/01/2021
11884                                  <1>       ; 11/01/2021 (TRDOS 386 v2.0.3)
11885                                  <1>       ; (vgabios.c)
11886                                  <1>
11887                                  <1>       ; modified registers: eax, edx, edi, ch
11888                                  <1>
11889                                  <1>       ;mov   edi, VBE3SAVERESTOREBLOCK
11890                                  <1>
11891                                  <1>       ; input: edi = state buffer address
11892                                  <1>
11893 00003BDD F6C101                  <1>       test   cl, 1
11894 00003BE0 0F8485000000            <1>       jz     bfn_svs_4
11895                                  <1>
11896 00003BE6 66BAC403                <1>       mov    dx, VGAREG_SEQU_ADDRESS ; 3C7h
11897 00003BEA EC                      <1>       in     al, dx
11898 00003BEB AA                      <1>       stosb
11899                                  <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
11900 00003BEC B2D4                    <1>       mov    dl, 0D4h
11901 00003BEE EC                      <1>       in     al, dx
11902 00003BEF AA                      <1>       stosb
11903                                  <1>       ;mov   dx, VGAREG_GRDC_ADDRESS ; 3CEh
11904 00003BF0 B2CE                    <1>         mov  dl, 0CEh
11905 00003BF2 EC                      <1>       in     al, dx
11906 00003BF3 AA                      <1>       stosb
11907                                  <1>       ;mov   dx, VGAREG_ACTL_RESET ; 3DAh
11908 00003BF4 B2DA                    <1>       mov    dl, 0DAh
11909 00003BF6 EC                      <1>       in     al, dx
11910                                  <1>       ;mov   dx, VGAREG_ACTL_ADDRESS ; 3C0h
11911 00003BF7 B2C0                    <1>       mov    dl, 0C0h
11912 00003BF9 EC                      <1>       in     al, dx
11913 00003BFA AA                      <1>       stosb
11914 00003BFB 88C4                    <1>       mov    ah, al ; ar_index
11915                                  <1>       ;mov   dx, VGAREG_READ_FEATURE_CTL ; 3CAh
11916 00003BFD B2CA                    <1>       mov    dl, 0CAh
11917 00003BFF EC                      <1>       in     al, dx
```

```
11918 00003C00 AA                <1>       stosb
11919                            <1>       ; (5 bytes are writen above)
11920                            <1>
11921                            <1>       ; for(i=1;i<=4;i++){
11922 00003C01 B001              <1>       mov   al, 1
11923                            <1>       ;;mov  dx, VGAREG_SEQU_ADDRESS ; 3C4h
11924                            <1>       ;mov   dl, 0C4h
11925 00003C03 B504              <1>       mov   ch, 4
11926                            <1> bfn_svs_0:
11927                            <1>       ; outb(VGAREG_SEQU_ADDRESS, i);
11928                            <1>       ;mov   dx, VGAREG_SEQU_ADDRESS ; 3C4h
11929 00003C05 B2C4              <1>       mov   dl, 0C4h
11930 00003C07 EE                <1>       out   dx, al
11931                            <1>       ;mov   dx, VGAREG_SEQU_DATA  ; 3C5h
11932 00003C08 FEC2              <1>       inc   dl  ; dx = 3C5h
11933                            <1>       ; inb(VGAREG_SEQU_DATA)
11934 00003C0A 50                <1>       push  eax
11935 00003C0B EC                <1>       in    al, dx
11936 00003C0C AA                <1>       stosb ; (4 bytes in loop)
11937 00003C0D 58                <1>       pop   eax
11938                            <1>       ;mov   dx, VGAREG_SEQU_ADDRESS ; 3C4h
11939                            <1>       ;dec   dl
11940 00003C0E FEC0              <1>       inc   al  ; i++
11941 00003C10 FECD              <1>       dec   ch
11942 00003C12 75F1              <1>       jnz   short bfn_svs_0
11943                            <1>
11944                            <1>       ; outb(VGAREG_SEQU_ADDRESS, 0);
11945 00003C14 28C0              <1>       sub   al, al ; 0
11946 00003C16 EE                <1>       out   dx, al
11947                            <1>       ; inb(VGAREG_SEQU_DATA)
11948                            <1>       ;mov   dx, VGAREG_SEQU_DATA ; 3C5h
11949 00003C17 FEC2              <1>       inc   dl  ; dx = 3C5h
11950 00003C19 EC                <1>       in    al, dx
11951 00003C1A AA                <1>       stosb ; (+1 byte)
11952                            <1>
11953                            <1>         ; for(i=0;i<=0x18;i++) {
11954 00003C1B 28C0              <1>       sub   al, al ; 0
11955                            <1>       ;;mov  dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
11956                            <1>       ;mov   dl, 0D4h
11957 00003C1D B519              <1>       mov   ch, 25
11958                            <1> bfn_svs_1:
11959                            <1>         ; outb(crtc_addr,i);
11960                            <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
11961 00003C1F B2D4              <1>       mov   dl, 0D4h
11962 00003C21 EE                <1>       out   dx, al
11963                            <1>       ;mov   dx, VGAREG_VGA_CRTC_DATA ; 3D5h
11964 00003C22 FEC2              <1>       inc   dl  ; dx = 3D5h
11965                            <1>       ; inb(crtc_addr+1)
11966 00003C24 50                <1>       push  eax
11967 00003C25 EC                <1>       in    al, dx
11968 00003C26 AA                <1>       stosb ; (25 bytes in loop)
11969 00003C27 58                <1>       pop   eax
11970                            <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
11971                            <1>       ;dec   dl
11972 00003C28 FEC0              <1>       inc   al  ; i++
11973 00003C2A FECD              <1>       dec   ch
11974 00003C2C 75F1              <1>       jnz   short bfn_svs_1
11975                            <1>
11976 00003C2E 80E420            <1>       and   ah, 20h  ; (ar_index & 0x20)
11977                            <1>         ; for(i=0;i<=0x13;i++) {
11978 00003C31 28C0              <1>       sub   al, al ; 0
11979 00003C33 B514              <1>       mov   ch, 20
11980                            <1> bfn_svs_2:
11981                            <1>       ; inb(VGAREG_ACTL_RESET);
11982                            <1>       ;mov   dx, VGAREG_ACTL_RESET ; 3DAh
11983 00003C35 B2DA              <1>       mov   dl, 0DAh
11984 00003C37 50                <1>       push  eax
11985 00003C38 EC                <1>       in    al, dx
11986 00003C39 8A0424            <1>       mov   al, [esp]
11987                            <1>       ; outb(VGAREG_ACTL_ADDRESS, i | (ar_index & 0x20));
11988 00003C3C 08E0              <1>       or    al, ah
11989                            <1>       ;mov   dx, VGAREG_ACTL_ADDRESS ; 3C0h
11990 00003C3E B2C0              <1>       mov   dl, 0C0h
11991 00003C40 EE                <1>       out   dx, al
11992                            <1>       ;mov   dx, VGAREG_ACTL_READ_DATA ; 3C1h
11993                            <1>       ;mov   dl, 0C1h
11994 00003C41 FEC2              <1>       inc   dl
11995 00003C43 EC                <1>       in    al, dx
11996 00003C44 AA                <1>       stosb ; (20 bytes in loop)
11997 00003C45 58                <1>       pop   eax
11998 00003C46 FEC0              <1>       inc   al  ; i++
11999 00003C48 FECD              <1>       dec   ch
12000 00003C4A 75E9              <1>       jnz   short bfn_svs_2
12001                            <1>
12002                            <1>       ; inb(VGAREG_ACTL_RESET);
12003                            <1>       ;mov   dx, VGAREG_ACTL_RESET ; 3DAh
12004 00003C4C B2DA              <1>       mov   dl, 0DAh
12005 00003C4E EC                <1>       in    al, dx
12006                            <1>
12007                            <1>         ; for(i=0;i<=8;i++) {
12008 00003C4F 28C0              <1>       sub   al, al ; 0
12009                            <1>       ;;mov  dx, VGAREG_GRDC_ADDRESS ; 3CEh
12010                            <1>       ;mov   dl, 0CEh
12011 00003C51 B509              <1>       mov   ch, 9
12012                            <1> bfn_svs_3:
12013                            <1>       ; outb(VGAREG_GRDC_ADDRESS,i)
12014                            <1>       ;mov   dx, VGAREG_GRDC_ADDRESS ; 3CEh
12015 00003C53 B2CE              <1>       mov   dl, 0CEh
12016 00003C55 EE                <1>       out   dx, al
12017                            <1>       ; inb(VGAREG_ACTL_READ_DATA)
12018 00003C56 50                <1>       push  eax
12019                            <1>       ;mov   dx, VGAREG_GRDC_DATA ; 3CFh
12020                            <1>       ;mov   dl, 0CFh
12021 00003C57 FEC2              <1>       inc   dl
12022 00003C59 EC                <1>       in    al, dx
```

```
12023 00003C5A AA                <1>       stosb  ; (9 bytes in loop)
12024 00003C5B 58                <1>       pop    eax
12025                            <1>       ;dec   dl
12026 00003C5C FEC0              <1>       inc    al  ; i++
12027 00003C5E FECD              <1>       dec    ch
12028 00003C60 75F1              <1>       jnz    short bfn_svs_3
12029                            <1>
12030                            <1>       ; write_word(ES, BX, crtc_addr); BX+= 2;
12031                            <1>       ; (offset 64)
12032 00003C62 66B8D403          <1>       mov    ax, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
12033 00003C66 66AB              <1>       stosw  ; (2 bytes (1 word))
12034                            <1>
12035                            <1>         ; /* XXX: read plane latches */
12036 00003C68 31C0              <1>       xor    eax, eax  ; 0
12037 00003C6A AB                <1>              stosd  ; (4 bytes)
12038                            <1>
12039                            <1>       ; (total 70 bytes are written above as controller hardware state)
12040                            <1>
12041                            <1> bfn_svs_4:
12042                            <1>       ; 12/01/2021 (TRDOS 386 v2.0.3)
12043 00003C6B F6C102            <1>       test   cl, 2
12044 00003C6E 7476              <1>       jz     short bfn_svs_6
12045                            <1>
12046                            <1>       ; VIDEO BIOS DATA
12047                            <1>       ; !!! this data is valid for TRDOS 386 v2 kernel only !!!
12048                            <1>       ; (this is not same with BOCHS/PLEX86 video bios, BIOS data)
12049                            <1>
12050                            <1>       ; if (CX & 2) {
12051                            <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_CURRENT_MODE)); BX++;
12052                            <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS)); BX += 2;
12053                            <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_PAGE_SIZE)); BX += 2;
12054                            <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_CRTC_ADDRESS)); BX += 2;
12055                            <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS)); BX++;
12056                            <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_CHAR_HEIGHT)); BX += 2;
12057                            <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_VIDEO_CTL)); BX++;
12058                            <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_SWITCHES)); BX++;
12059                            <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_MODESET_CTL)); BX++;
12060                            <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_CURSOR_TYPE)); BX += 2;
12061                            <1>         ;for(i=0;i<8;i++) {
12062                            <1>         ;   write_word(ES, BX, read_word(BIOSMEM_SEG, BIOSMEM_CURSOR_POS+2*i));
12063                            <1>         ;   BX += 2;
12064                            <1>         ;}
12065                            <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_CURRENT_START)); BX += 2;
12066                            <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_CURRENT_PAGE)); BX++;
12067                            <1>         ;/* current font */
12068                            <1>         ;write_word(ES, BX, read_word(0, 0x1f * 4)); BX += 2;
12069                            <1>         ;write_word(ES, BX, read_word(0, 0x1f * 4 + 2)); BX += 2;
12070                            <1>         ;write_word(ES, BX, read_word(0, 0x43 * 4)); BX += 2;
12071                            <1>         ;write_word(ES, BX, read_word(0, 0x43 * 4 + 2)); BX += 2;
12072                            <1>
12073                            <1>       ; !!! save TRDOS 386 v2 kernel spesific video bios data !!!
12074                            <1>       ; (which is/are used by 'SET_MODE' function and/or it's sub functions)
12075                            <1>
12076 00003C70 66B8D403          <1>       mov    ax, 3D4h ; CRTC_ADDR, always 3D4h (color VGA) for TRDOS 386 v2
12077 00003C74 66AB              <1>       stosw
12078 00003C76 A0[BA6A0000]      <1>       mov    al, [CRT_MODE] ; Current video mode (0FFh for VESA VBE modes)
12079 00003C7B AA                <1>       stosb
12080 00003C7C A0[BB6A0000]      <1>       mov    al, [CRT_MODE_SET] ; 29h for mode 03h ; TRDOS 386 feature only !
12081 00003C81 AA                <1>       stosb
12082 00003C82 66A1[06120300]    <1>       mov    ax, [video_mode] ; Current VESA VBE (SVGA, extended VGA) mode
12083 00003C88 66AB              <1>       stosw                   ; (valid if [CRT_MODE] = 0FFh)
12084 00003C8A 66A1[408D0100]    <1>       mov    ax, [CRT_LEN] ; page size (in bytes)
12085 00003C90 66AB              <1>       stosw
12086 00003C92 66A1[C4800100]    <1>       mov    ax, [CRT_START] ; video page start offset
12087 00003C98 66AB              <1>       stosw
12088 00003C9A A0[BC6A0000]      <1>       mov    al, [CRT_COLS] ; nbcols, characters per row
12089 00003C9F AA                <1>       stosb
12090 00003CA0 A0[C26A0000]      <1>       mov    al, [VGA_ROWS] ; nbrows, (character) rows per page (not rows-1)
12091 00003CA5 AA                <1>       stosb
12092 00003CA6 A0[BE6A0000]      <1>       mov    al, [CHAR_HEIGHT] ; character font height (8 or 16 or 14)
12093 00003CAB AA                <1>       stosb
12094 00003CAC A0[BF6A0000]      <1>       mov    al, [VGA_VIDEO_CTL] ; ROM BIOS DATA AREA Offset 87h
12095 00003CB1 AA                <1>       stosb
12096 00003CB2 A0[C06A0000]      <1>       mov    al, [VGA_SWITCHES] ; feature bit switches
12097 00003CB7 AA                <1>       stosb
12098 00003CB8 A0[C16A0000]      <1>       mov    al, [VGA_MODESET_CTL] ; basic mode set options
12099 00003CBD AA                <1>       stosb
12100                            <1>       ; followings are only used by TRDOS 386 v2 (IBM PC/AT ROMBIOS) code
12101                            <1>       ; (bochs/plex86 does not use and return those)
12102 00003CBE A0[BD6A0000]      <1>       mov    al, [CRT_PALETTE] ; current color palette ; TRDOS 386 feature only !
12103 00003CC3 AA                <1>       stosb
12104 00003CC4 A0[D6800100]      <1>       mov    al, [ACTIVE_PAGE] ; current video page
12105 00003CC9 AA                <1>       stosb
12106 00003CCA 66A1[D36A0000]    <1>       mov    ax, [CURSOR_MODE] ; cursor type
12107 00003CD0 66AB              <1>       stosw
12108                            <1>       ;mov   eax, [CURSOR_POSN] ; cursor position for video page 0 and 1
12109                            <1>       ;stosd
12110                            <1>       ;mov   eax, [CURSOR_POSN+4] ; cursor position for video page 2 and 3
12111                            <1>       ;stosd
12112                            <1>       ;mov   eax, [CURSOR_POSN+8] ; cursor position for video page 4 and 5
12113                            <1>       ;stosd
12114                            <1>       ;mov   eax, [CURSOR_POSN+12] ; cursor position for video page 6 and 7
12115                            <1>       ;stosd
12116 00003CD2 56                <1>       push   esi
12117 00003CD3 B504              <1>       mov    ch, 4
12118 00003CD5 BE[C6800100]      <1>       mov    esi, CURSOR_POSN
12119                            <1> bfn_svs_5:
12120 00003CDA A5                <1>       movsd
12121 00003CDB FECD              <1>       dec    ch
12122 00003CDD 75FB              <1>       jnz    short bfn_svs_5
12123 00003CDF 5E                <1>       pop    esi
12124                            <1>       ; (font addr) protected mode address in kernel's/system memory space
12125                            <1>       ; (not accessable/meaningful address value by user)
12126 00003CE0 A1[528D0100]      <1>       mov    eax, [VGA_INT43H] ; VGA current (default) font address
12127 00003CE5 AB                <1>       stosd
```

```
                                <1>
                                <1>       ; (total 40 bytes are written above as BIOS data state)
                                <1>
                                <1> bfn_svs_6:
                                <1>       ; 12/01/2021
00003CE6 F6C104                 <1>       test   cl, 4
00003CE9 7423                   <1>       jz     short bfn_svs_8
                                <1>
                                <1>       ;/* XXX: check this */
                                <1>              ; /* read/write mode dac */
                                <1>         ;write_byte(ES, BX, inb(VGAREG_DAC_STATE)); BX++;
                                <1>       ;      ; /* pix address */
                                <1>         ;write_byte(ES, BX, inb(VGAREG_DAC_WRITE_ADDRESS)); BX++;
                                <1>         ;write_byte(ES, BX, inb(VGAREG_PEL_MASK)); BX++;
                                <1>         ;// Set the whole dac always, from 0
                                <1>         ;outb(VGAREG_DAC_WRITE_ADDRESS,0x00);
                                <1>         ;for(i=0;i<256*3;i++) {
                                <1>         ;   write_byte(ES, BX, inb(VGAREG_DAC_DATA)); BX++;
                                <1>         ;}
                                <1>         ;write_byte(ES, BX, 0); BX++; /* color select register */
                                <1>
                                <1>       ; /* read/write mode dac */
00003CEB 66BAC703               <1>       mov    dx, 3C7h ; VGAREG_DAC_STATE
00003CEF EC                     <1>       in     al, dx
00003CF0 AA                     <1>       stosb
                                <1>       ; /* pix address */
                                <1>       ;mov   dx, VGAREG_DAC_WRITE_ADDRESS ; 3C8h
                                <1>       ;mov   dl, 0C8h
00003CF1 FEC2                   <1>       inc    dl
00003CF3 EC                     <1>       in     al, dx
00003CF4 AA                     <1>       stosb
                                <1>       ;mov   dx, VGAREG_PEL_MASK  ; 3C6h
00003CF5 B2C6                   <1>       mov    dl, 0C6h
00003CF7 EC                     <1>       in     al, dx
00003CF8 AA                     <1>       stosb
                                <1>       ;// Set the whole dac always, from 0
00003CF9 30C0                   <1>       xor    al, al ; 0
                                <1>       ;mov   dx, VGAREG_DAC_WRITE_ADDRESS ; 3C8h
00003CFB B2C8                   <1>       mov    dl, 0C8h
00003CFD EE                     <1>       out    dx, al
                                <1>
00003CFE 51                     <1>       push   ecx ; 22/01/2021
                                <1>       ;for(i=0;i<256*3;i++) {
00003CFF B900030000             <1>       mov    ecx, 256*3 ; 768 bytes
                                <1>       ;mov   dx, VGAREG_DAC_DATA ; 3C9h
                                <1>       ;mov   dl, 0C9h
00003D04 FEC2                   <1>       inc    dl ; dx = 3C9h
                                <1> bfn_svs_7:
00003D06 EC                     <1>       in     al, dx
00003D07 AA                     <1>       stosb
00003D08 E2FC                   <1>       loop   bfn_svs_7
00003D0A 59                     <1>       pop    ecx ; 22/01/2021
                                <1>
                                <1>       ; /* color select register */
00003D0B 28C0                   <1>       sub    al, al ; 0
00003D0D AA                     <1>       stosb
                                <1>
                                <1>       ; (total 772 bytes are written above as DAC state)
                                <1> bfn_svs_8:
00003D0E C3                     <1>       retn
                                <1>
                                <1> vbe_biosfn_save_video_state:
                                <1>       ; 23/01/2021
                                <1>       ; 13/01/2021
                                <1>       ; 12/01/2021 (TRDOS 386 v2.0.3)
                                <1>       ; (vbe.c)
                                <1>
                                <1>       ; modified registers: eax, edx, edi, ch
                                <1>
                                <1>       ; input: edi = state buffer address
                                <1>       ; output:
                                <1>       ;      VBE DISPI register contents will be saved
                                <1>       ;      (18 bytes, 9 words)
                                <1>
                                <1>       ; outw(VBE_DISPI_IOPORT_INDEX,VBE_DISPI_INDEX_ENABLE);
                                <1>       ; enable = inw(VBE_DISPI_IOPORT_DATA);
                                <1>       ; write_word(ES, BX, enable);
                                <1>       ; BX += 2;
                                <1>       ; if (!(enable & VBE_DISPI_ENABLED))
                                <1>       ;     return;
                                <1>       ; for(i = VBE_DISPI_INDEX_XRES;
                                <1>       ;             i <= VBE_DISPI_INDEX_Y_OFFSET; i++) {
                                <1>       ;    if (i != VBE_DISPI_INDEX_ENABLE) {
                                <1>       ;        outw(VBE_DISPI_IOPORT_INDEX, i);
                                <1>       ;        write_word(ES, BX, inw(VBE_DISPI_IOPORT_DATA));
                                <1>         ;    BX += 2;
                                <1>       ;        }
                                <1>       ; }
                                <1>
00003D0F 66BACE01               <1>       mov    dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
00003D13 B804000000             <1>       mov    eax, 04h  ; VBE_DISPI_INDEX_ENABLE
00003D18 66EF                   <1>       out    dx, ax
                                <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
00003D1A FEC2                   <1>       inc    dl
00003D1C 66ED                   <1>       in     ax, dx ; enable (status)
00003D1E 66AB                   <1>       stosw
00003D20 6683E001               <1>       and    ax, 1 ; VBE_DISPI_ENABLED
00003D24 7505                   <1>       jnz    short vbe_bfn_svs_0
                                <1>       ; 23/01/2021
                                <1>       ; ax = 0
                                <1>       ; VBE_DISPI_DISABLED
                                <1>       ; 13/01/2021
                                <1>       ; clear remain 8 bytes
                                <1>       ;xor   eax, eax
00003D26 AB                     <1>       stosd  ; 2
```

```
12233 00003D27 AB                <1>       stosd  ; 2
12234 00003D28 AB                <1>       stosd  ; 2
12235 00003D29 AB                <1>       stosd  ; 2
12236 00003D2A C3                <1>       retn
12237                            <1> vbe_bfn_svs_0:
12238                            <1>       ; VBE_DISPI_ENABLED
12239                            <1>
12240                            <1>       ;sub   eax, eax
12241 00003D2B 28C0              <1>       sub    al, al ; eax = 0
12242                            <1>
12243                            <1>       ; from VBE_DISPI_INDEX_XRES
12244                            <1>       ;   to VBE_DISPI_INDEX_BPP
12245                            <1>
12246 00003D2D B503              <1>       mov    ch, 3
12247                            <1>       ; al = 0 ; VBE_DISPI_INDEX_XRES - 1
12248                            <1>
12249 00003D2F E804000000        <1>       call   vbe_bfn_svs_1
12250                            <1>
12251                            <1>       ; from VBE_DISPI_INDEX_BANK
12252                            <1>       ;   to VBE_DISPI_INDEX_Y_OFFSET
12253                            <1>
12254 00003D34 FEC0              <1>       inc    al
12255                            <1>       ; al = 4 ; VBE_DISPI_INDEX_BANK - 1
12256                            <1>
12257 00003D36 B505              <1>       mov    ch, 5
12258                            <1> vbe_bfn_svs_1:
12259 00003D38 FEC0              <1>       inc    al ; from VBE_DISPI_INDEX_XRES
12260                            <1>                  ;   to VBE_DISPI_INDEX_BPP
12261                            <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
12262 00003D3A FECA              <1>       dec    dl ; 1CEh
12263 00003D3C 66EF              <1>       out    dx, ax
12264 00003D3E 50                <1>       push   eax
12265                            <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
12266 00003D3F FEC2              <1>       inc    dl ; 1CFh
12267 00003D41 66ED              <1>       in     ax, dx
12268 00003D43 66AB              <1>       stosw
12269 00003D45 58                <1>       pop    eax
12270 00003D46 FECD              <1>       dec    ch
12271 00003D48 75EE              <1>       jnz    short vbe_bfn_svs_1
12272 00003D4A C3                <1>       retn
12273                            <1>
12274                            <1> ; 11/01/2021
12275                            <1> VGAREG_ACTL_ADDRESS       equ 3C0h
12276                            <1> VGAREG_ACTL_WRITE_DATA    equ 3C0h
12277                            <1> VGAREG_ACTL_READ_DATA     equ 3C1h
12278                            <1>
12279                            <1> VGAREG_INPUT_STATUS       equ 3C2h
12280                            <1> VGAREG_WRITE_MISC_OUTPUT equ 3C2h
12281                            <1> VGAREG_VIDEO_ENABLE       equ 3C3h
12282                            <1> VGAREG_SEQU_ADDRESS       equ 3C4h
12283                            <1> VGAREG_SEQU_DATA   equ 3C5h
12284                            <1>
12285                            <1> VGAREG_PEL_MASK           equ 3C6h
12286                            <1> VGAREG_DAC_STATE   equ 3C7h
12287                            <1> VGAREG_DAC_READ_ADDRESS   equ 3C7h
12288                            <1> VGAREG_DAC_WRITE_ADDRESS equ 3C8h
12289                            <1> VGAREG_DAC_DATA           equ 3C9h
12290                            <1>
12291                            <1> VGAREG_READ_FEATURE_CTL   equ 3CAh
12292                            <1> VGAREG_READ_MISC_OUTPUT   equ 3CCh
12293                            <1>
12294                            <1> VGAREG_GRDC_ADDRESS       equ 3CEh
12295                            <1> VGAREG_GRDC_DATA   equ 3CFh
12296                            <1>
12297                            <1> ;VGAREG_MDA_CRTC_ADDRESS equ 3B4h
12298                            <1> ;VGAREG_MDA_CRTC_DATA     equ 3B5h
12299                            <1> VGAREG_VGA_CRTC_ADDRESS   equ 3D4h
12300                            <1> VGAREG_VGA_CRTC_DATA      equ 3D5h
12301                            <1>
12302                            <1> ;VGAREG_MDA_WRITE_FEATURE_CTL equ 3BAh
12303                            <1> VGAREG_VGA_WRITE_FEATURE_CTL equ 3DAh
12304                            <1> VGAREG_ACTL_RESET  equ 3DAh
12305                            <1>
12306                            <1> ;VGAREG_MDA_MODECTL       equ 3B8h
12307                            <1> VGAREG_CGA_MODECTL equ 3D8h
12308                            <1> VGAREG_CGA_PALETTE equ 3D9h
12309                            <1>
12310                            <1> biosfn_restore_video_state:
12311                            <1>       ; 22/01/2021
12312                            <1>       ; 13/01/2021
12313                            <1>       ; 12/01/2021 (TRDOS 386 v2.0.3)
12314                            <1>       ; (vgabios.c)
12315                            <1>
12316                            <1>       ; modified registers: eax, edx, esi, edi, ch
12317                            <1>
12318                            <1>       ;mov   esi, VBE3SAVERESTOREBLOCK
12319                            <1>
12320                            <1>       ; input: esi = state buffer address
12321                            <1>
12322 00003D4B F6C101            <1>       test   cl, 1
12323 00003D4E 0F84A9000000      <1>       jz     bfn_rvs_6
12324                            <1>
12325 00003D54 66817E40D403      <1>       cmp    word [esi+64], 3D4h ; must be 3D4h
12326 00003D5A 7402              <1>       je     short bfn_rvs_0
12327                            <1>                     ; it is seen as valid buffer
12328 00003D5C F9                <1>       stc
12329 00003D5D C3                <1>       retn
12330                            <1>
12331                            <1> bfn_rvs_0:
12332 00003D5E 89F7              <1>       mov    edi, esi ; addr1
12333 00003D60 83C605            <1>       add    esi, 5 ; skip 1st 5 bytes for now
12334                            <1>
12335                            <1>       ; // Reset Attribute Ctl flip-flop
12336                            <1>         ; inb(VGAREG_ACTL_RESET);
12337 00003D63 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
```

```
12338 00003D67 EC                <1>       in     al, dx
12339                            <1>
12340                            <1>       ; for(i=1;i<=4;i++){
12341 00003D68 B001              <1>       mov    al, 1
12342                            <1>       ;;mov  dx, VGAREG_SEQU_ADDRESS ; 3C4h
12343                            <1>       ;mov   dl, 0C4h
12344 00003D6A B504              <1>       mov    ch, 4
12345                            <1> bfn_rvs_1:
12346                            <1>       ; outb(VGAREG_SEQU_ADDRESS, i);
12347                            <1>       ;mov   dx, VGAREG_SEQU_ADDRESS ; 3C4h
12348 00003D6C B2C4              <1>       mov    dl, 0C4h
12349 00003D6E EE                <1>       out    dx, al
12350                            <1>       ;mov   dx, VGAREG_SEQU_DATA  ; 3C5h
12351 00003D6F FEC2              <1>       inc    dl  ; dx = 3C5h
12352                            <1>       ; outb(VGAREG_SEQU_DATA)
12353 00003D71 50                <1>       push   eax
12354 00003D72 AC                <1>       lodsb  ; (4 bytes in loop)
12355 00003D73 EE                <1>       out    dx, al
12356 00003D74 58                <1>       pop    eax
12357                            <1>       ;mov   dx, VGAREG_SEQU_ADDRESS ; 3C4h
12358                            <1>       ;dec   dl
12359 00003D75 FEC0              <1>       inc    al  ; i++
12360 00003D77 FECD              <1>       dec    ch
12361 00003D79 75F1              <1>       jnz    short bfn_rvs_1
12362                            <1>
12363                            <1>       ; outb(VGAREG_SEQU_ADDRESS, 0);
12364 00003D7B 28C0              <1>       sub    al, al ; 0
12365 00003D7D EE                <1>       out    dx, al
12366                            <1>       ; outb(VGAREG_SEQU_DATA)
12367                            <1>       ;mov   dx, VGAREG_SEQU_DATA ; 3C5h
12368 00003D7E FEC2              <1>       inc    dl  ; dx = 3C5h
12369 00003D80 AC                <1>       lodsb  ; (+1 byte)
12370 00003D81 EE                <1>       out    dx, al
12371                            <1>
12372                            <1>       ; // Disable CRTC write protection
12373                            <1>       ; outw(crtc_addr,0x0011);
12374                            <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
12375 00003D82 B2D4              <1>       mov    dl, 0D4h
12376 00003D84 66B81100          <1>       mov    ax, 11h
12377 00003D88 66EF              <1>       out    dx, ax
12378                            <1>
12379                            <1>       ; // Set CRTC regs
12380                            <1>
12381                            <1>       ; for(i=0;i<=0x18;i++) {
12382                            <1>         ;   if (i != 0x11) {
12383 00003D8A 28C0              <1>       sub    al, al ; 0
12384                            <1>       ;;mov  dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
12385                            <1>       ;mov   dl, 0D4h
12386 00003D8C B519              <1>       mov    ch, 25
12387                            <1> bfn_rvs_2:
12388                            <1>         ; outb(crtc_addr,i);
12389                            <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
12390 00003D8E B2D4              <1>       mov    dl, 0D4h
12391 00003D90 EE                <1>       out    dx, al
12392                            <1>       ;mov   dx, VGAREG_VGA_CRTC_DATA ; 3D5h
12393 00003D91 FEC2              <1>       inc    dl  ; dx = 3D5h
12394                            <1>       ; inb(crtc_addr+1)
12395 00003D93 50                <1>       push   eax
12396 00003D94 AC                <1>       lodsb  ; (25 bytes in loop)
12397 00003D95 EE                <1>       out    dx, al
12398 00003D96 58                <1>       pop    eax
12399                            <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
12400                            <1>       ;dec   dl
12401 00003D97 FEC0              <1>       inc    al  ; i++
12402 00003D99 3C11              <1>       cmp    al, 17 ; 11h
12403 00003D9B 7505              <1>       jne    short bfn_rvs_3
12404 00003D9D AC                <1>       lodsb
12405 00003D9E 88C4              <1>       mov    ah, al ; *
12406 00003DA0 B012              <1>       mov    al, 18
12407                            <1> bfn_rvs_3:
12408 00003DA2 FECD              <1>       dec    ch
12409 00003DA4 75E8              <1>       jnz    short bfn_rvs_2
12410                            <1>
12411                            <1>       ; // select crtc base address
12412                            <1>         ; v = inb(VGAREG_READ_MISC_OUTPUT) & ~0x01;
12413                            <1>       ;if (crtc_addr = 0x3d4)
12414                            <1>       ;   v |= 0x01;
12415                            <1>       ; outb(VGAREG_WRITE_MISC_OUTPUT, v);
12416                            <1>
12417                            <1>       ;;mov  dx, VGAREG_READ_MISC_OUTPUT ; 3CCh
12418                            <1>       ;mov   dl, 0CCh
12419                            <1>       ;in    al, dl
12420                            <1>       ;and   al, 1
12421                            <1>       ;;mov  dx, VGAREG_WRITE_MISC_OUTPUT ; 3C2h
12422                            <1>       ;mov   dl, 0C2h
12423                            <1>       ;or    al, 1
12424                            <1>       ;out   dx, al
12425                            <1>
12426                            <1>       ; // enable write protection if needed
12427                            <1>       ;outb(crtc_addr, 0x11);
12428                            <1>         ;outb(crtc_addr+1, read_byte(ES, BX - 0x18 + 0x11));
12429                            <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
12430 00003DA6 B2D4              <1>       mov    dl, 0D4h
12431 00003DA8 B011              <1>       mov    al, 11h
12432 00003DAA EE                <1>       out    dx, al
12433 00003DAB 88E0              <1>       mov    al, ah ; *
12434 00003DAD FEC2              <1>       inc    dl ; dx = 3D5h
12435 00003DAF EE                <1>       out    dx, al
12436                            <1>
12437                            <1>       ; // Set Attribute Ctl
12438 00003DB0 8A6703            <1>       mov    ah, [edi+3] ; addr1+3, ah = ar_index
12439 00003DB3 80E420            <1>       and    ah, 20h  ; (ar_index & 0x20)
12440                            <1>
12441                            <1>         ; inb(VGAREG_ACTL_RESET);
12442                            <1>       ;mov   dx, 3DAh ; VGAREG_ACTL_RESET
```

```
12443 00003DB6 B2DA              <1>       mov   dl, 0DAh
12444 00003DB8 EC                <1>       in    al, dx
12445                            <1>
12446                            <1>         ; for(i=0;i<=0x13;i++) {
12447 00003DB9 28C0              <1>       sub   al, al ; 0
12448 00003DBB B514              <1>       mov   ch, 20
12449                            <1> bfn_rvs_4:
12450                            <1>       ; outb(VGAREG_ACTL_ADDRESS, i | (ar_index & 0x20));
12451 00003DBD 50                <1>       push  eax
12452 00003DBE 08E0              <1>       or    al, ah
12453                            <1>       ;mov  dx, VGAREG_ACTL_ADDRESS ; 3C0h
12454 00003DC0 B2C0              <1>       mov   dl, 0C0h
12455 00003DC2 EE                <1>       out   dx, al
12456                            <1>       ;mov  dx, VGAREG_ACTL_WRITE_DATA ; 3C0h
12457                            <1>       ;mov  dl, 0C0h
12458 00003DC3 AC                <1>       lodsb ; (20 bytes in loop)
12459 00003DC4 EE                <1>       out   dx, al
12460 00003DC5 58                <1>       pop   eax
12461 00003DC6 FEC0              <1>       inc   al  ; i++
12462 00003DC8 FECD              <1>       dec   ch
12463 00003DCA 75F1              <1>       jnz   short bfn_rvs_4
12464                            <1>
12465                            <1>       ; outb(VGAREG_ACTL_ADDRESS, ar_index);
12466                            <1>       ;mov  dx, VGAREG_ACTL_ADDRESS ; 3C0h
12467                            <1>       ;mov  dl, 0C0h
12468 00003DCC 88E0              <1>       mov   al, ah ; ar_index
12469 00003DCE EE                <1>       out   dx, al
12470                            <1>
12471                            <1>       ; inb(VGAREG_ACTL_RESET);
12472                            <1>       ;mov  dx, VGAREG_ACTL_RESET ; 3DAh
12473 00003DCF B2DA              <1>       mov   dl, 0DAh
12474 00003DD1 EC                <1>       in    al, dx
12475                            <1>
12476                            <1>         ; for(i=0;i<=8;i++) {
12477 00003DD2 28C0              <1>       sub   al, al ; 0
12478                            <1>       ;;mov dx, VGAREG_GRDC_ADDRESS ; 3CEh
12479                            <1>       ;mov  dl, 0CEh
12480 00003DD4 B509              <1>       mov   ch, 9
12481                            <1> bfn_rvs_5:
12482                            <1>       ; outb(VGAREG_GRDC_ADDRESS,i)
12483                            <1>       ;mov  dx, VGAREG_GRDC_ADDRESS ; 3CEh
12484 00003DD6 B2CE              <1>       mov   dl, 0CEh
12485 00003DD8 EE                <1>       out   dx, al
12486                            <1>       ; outb(VGAREG_ACTL_READ_DATA)
12487 00003DD9 50                <1>       push  eax
12488                            <1>       ;mov  dx, VGAREG_GRDC_DATA ; 3CFh
12489                            <1>       ;mov  dl, 0CFh
12490 00003DDA FEC2              <1>       inc   dl
12491 00003DDC AC                <1>       lodsb ; (9 bytes in loop)
12492 00003DDD EE                <1>       out   dx, al
12493 00003DDE 58                <1>       pop   eax
12494                            <1>       ;dec  dl
12495 00003DDF FEC0              <1>       inc   al  ; i++
12496 00003DE1 FECD              <1>       dec   ch
12497 00003DE3 75F1              <1>       jnz   short bfn_rvs_5
12498                            <1>
12499                            <1>       ; BX += 2; /* crtc_addr */     ; 3D4h
12500                            <1>         ; BX += 4; /* plane latches */ ; 0
12501 00003DE5 83C606            <1>       add   esi, 6
12502 00003DE8 56                <1>       push  esi ; *
12503                            <1>
12504                            <1>       ;outb(VGAREG_SEQU_ADDRESS, read_byte(ES, addr1)); addr1++;
12505                            <1>         ;outb(crtc_addr, read_byte(ES, addr1)); addr1++;
12506                            <1>         ;outb(VGAREG_GRDC_ADDRESS, read_byte(ES, addr1)); addr1++;
12507                            <1>         ;addr1++;
12508                            <1>         ;outb(crtc_addr - 0x4 + 0xa, read_byte(ES, addr1)); addr1++;
12509                            <1>
12510 00003DE9 89FE              <1>       mov   esi, edi ; start of state buffer
12511                            <1>
12512                            <1>       ;mov  dx, VGAREG_SEQU_ADDRESS ; 3C7h
12513 00003DEB B2C7              <1>       mov   dl, 0C7h
12514 00003DED AC                <1>       lodsb
12515 00003DEE EE                <1>       out   dx, al
12516                            <1>       ;mov  dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
12517 00003DEF B2D4              <1>       mov   dl, 0D4h
12518 00003DF1 AC                <1>       lodsb
12519 00003DF2 EE                <1>       out   dx, al
12520                            <1>       ;mov  dx, VGAREG_GRDC_ADDRESS ; 3CEh
12521 00003DF3 B2CE              <1>         mov dl, 0CEh
12522 00003DF5 AC                <1>       lodsb
12523 00003DF6 EE                <1>       out   dx, al
12524 00003DF7 AC                <1>       lodsb ; addr1++
12525                            <1>       ;mov  dx, VGAREG_VGA_WRITE_FEATURE_CTL ; 3DAh
12526 00003DF8 B2DA              <1>       mov   dl, 0DAh
12527 00003DFA AC                <1>       lodsb
12528 00003DFB EE                <1>       out   dx, al
12529                            <1>
12530 00003DFC 5E                <1>       pop   esi ; *
12531                            <1>
12532                            <1>       ; (total 70 bytes are read above as controller hardware state)
12533                            <1>
12534                            <1> bfn_rvs_6:
12535                            <1>       ; 13/01/2021
12536 00003DFD F6C102            <1>       test  cl, 2
12537 00003E00 747D              <1>       jz    short bfn_rvs_9
12538                            <1>
12539                            <1>       ; VIDEO BIOS DATA
12540                            <1>       ; !!! this data is valid for TRDOS 386 v2 kernel only !!!
12541                            <1>       ; (this is not same with BOCHS/PLEX86 video bios, BIOS data)
12542                            <1>
12543                            <1>       ; if (CX & 2) {
12544                            <1>       ;write_byte(BIOSMEM_SEG,BIOSMEM_CURRENT_MODE, read_byte(ES, BX)); BX++;
12545                            <1>         ;write_word(BIOSMEM_SEG,BIOSMEM_NB_COLS, read_word(ES, BX)); BX += 2;
12546                            <1>         ;write_word(BIOSMEM_SEG,BIOSMEM_PAGE_SIZE, read_word(ES, BX)); BX += 2;
12547                            <1>         ;write_word(BIOSMEM_SEG,BIOSMEM_CRTC_ADDRESS, read_word(ES, BX)); BX += 2;
```

```
12548                                  <1>         ;write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, read_byte(ES, BX)); BX++;
12549                                  <1>         ;write_word(BIOSMEM_SEG,BIOSMEM_CHAR_HEIGHT, read_word(ES, BX)); BX += 2;
12550                                  <1>         ;write_byte(BIOSMEM_SEG,BIOSMEM_VIDEO_CTL, read_byte(ES, BX)); BX++;
12551                                  <1>         ;write_byte(BIOSMEM_SEG,BIOSMEM_SWITCHES, read_byte(ES, BX)); BX++;
12552                                  <1>         ;write_byte(BIOSMEM_SEG,BIOSMEM_MODESET_CTL, read_byte(ES, BX)); BX++;
12553                                  <1>         ;write_word(BIOSMEM_SEG,BIOSMEM_CURSOR_TYPE, read_word(ES, BX)); BX += 2;
12554                                  <1>         ;for(i=0;i<8;i++) {
12555                                  <1>         ;   write_word(BIOSMEM_SEG, BIOSMEM_CURSOR_POS+2*i, read_word(ES, BX));
12556                                  <1>         ;   BX += 2;
12557                                  <1>         ;}
12558                                  <1>         ;write_word(BIOSMEM_SEG,BIOSMEM_CURRENT_START, read_word(ES, BX)); BX += 2;
12559                                  <1>         ;write_byte(BIOSMEM_SEG,BIOSMEM_CURRENT_PAGE, read_byte(ES, BX)); BX++;
12560                                  <1>         ;/* current font */
12561                                  <1>         ;write_word(0, 0x1f * 4, read_word(ES, BX)); BX += 2;
12562                                  <1>         ;write_word(0, 0x1f * 4 + 2, read_word(ES, BX)); BX += 2;
12563                                  <1>         ;write_word(0, 0x43 * 4, read_word(ES, BX)); BX += 2;
12564                                  <1>         ;write_word(0, 0x43 * 4 + 2, read_word(ES, BX)); BX += 2;
12565                                  <1>
12566                                  <1>       ; !!! save TRDOS 386 v2 kernel spesific video bios data !!!
12567                                  <1>       ; (which is/are used by 'SET_MODE' function and/or it's sub functions)
12568                                  <1>
12569 00003E02 66AD                  <1>       lodsw   ; CRTC_ADDR, always 3D4h (color VGA) for TRDOS 386 v2
12570                                  <1>       ; skip 3D4h check if it is already checked
12571 00003E04 F6C101                <1>       test   cl, 1
12572 00003E07 7508                  <1>       jnz    short bfn_rvs_7
12573 00003E09 663DD403              <1>       cmp    ax, 3D4h
12574 00003E0D 7402                  <1>       je     short bfn_rvs_7
12575 00003E0F F9                    <1>       stc
12576 00003E10 C3                    <1>       retn
12577                                <1> bfn_rvs_7:
12578 00003E11 AC                    <1>       lodsb
12579 00003E12 A2[BA6A0000]          <1>       mov    [CRT_MODE], al ; Current video mode (0FFh for VESA VBE modes)
12580 00003E17 AC                    <1>       lodsb
12581 00003E18 A2[BB6A0000]          <1>       mov    [CRT_MODE_SET], al ; 29h for mode 03h ; TRDOS 386 feature only !
12582 00003E1D 66AD                  <1>       lodsw
12583 00003E1F 66A3[06120300]        <1>       mov    [video_mode], ax ; Current VESA VBE (SVGA, extended VGA) mode
12584 00003E25 66AD                  <1>       lodsw          ; (valid if [CRT_MODE] = 0FFh)
12585 00003E27 66A3[408D0100]        <1>       mov    [CRT_LEN], ax ; page size (in bytes)
12586 00003E2D 66AD                  <1>       lodsw
12587 00003E2F 66A3[C4800100]        <1>       mov    [CRT_START], ax ; video page start offset
12588 00003E35 AC                    <1>       lodsb
12589 00003E36 A2[BC6A0000]          <1>       mov    [CRT_COLS], al ; nbcols, characters per row
12590 00003E3B AC                    <1>       lodsb
12591 00003E3C A2[C26A0000]          <1>       mov    [VGA_ROWS], al ; nbrows, (character) rows per page (not rows-1)
12592 00003E41 AC                    <1>       lodsb
12593 00003E42 A2[BE6A0000]          <1>       mov    [CHAR_HEIGHT], al ; character font height (8 or 16 or 14)
12594 00003E47 AC                    <1>       lodsb
12595 00003E48 A2[BF6A0000]          <1>       mov    [VGA_VIDEO_CTL], al ; ROM BIOS DATA AREA Offset 87h
12596 00003E4D AC                    <1>       lodsb
12597 00003E4E A2[C06A0000]          <1>       mov    [VGA_SWITCHES], al ; feature bit switches
12598 00003E53 AC                    <1>       lodsb
12599 00003E54 A2[C16A0000]          <1>       mov    [VGA_MODESET_CTL], al ; basic mode set options
12600                                <1>       ; followings are only used by TRDOS 386 v2 (IBM PC/AT ROMBIOS) code
12601                                <1>       ; (bochs/plex86 does not use and return those)
12602 00003E59 AC                    <1>       lodsb
12603 00003E5A A2[BD6A0000]          <1>       mov    [CRT_PALETTE], al ; current color palette ; TRDOS 386 feature only !
12604 00003E5F AC                    <1>       lodsb
12605 00003E60 A2[D6800100]          <1>       mov    [ACTIVE_PAGE], al ; current video page
12606 00003E65 66AD                  <1>       lodsw
12607 00003E67 66A3[D36A0000]        <1>       mov    [CURSOR_MODE], ax ; cursor type
12608                                <1>       ;lodsd
12609                                <1>       ;mov   [CURSOR_POSN], eax ; cursor position for video page 0 and 1
12610                                <1>       ;lodsd
12611                                <1>       ;mov   [CURSOR_POSN+4], eax ; cursor position for video page 2 and 3
12612                                <1>       ;lodsd
12613                                <1>       ;mov   [CURSOR_POSN+8], eax ; cursor position for video page 4 and 5
12614                                <1>       ;lodsd
12615                                <1>       ;mov   [CURSOR_POSN+12], eax ; cursor position for video page 6 and 7
12616 00003E6D B504                  <1>       mov    ch, 4
12617 00003E6F BF[C6800100]          <1>       mov    edi, CURSOR_POSN
12618                                <1> bfn_rvs_8:
12619 00003E74 A5                    <1>       movsd
12620 00003E75 FECD                  <1>       dec    ch
12621 00003E77 75FB                  <1>       jnz    short bfn_rvs_8
12622                                <1>       ; (font addr) protected mode address in kernel's/system memory space
12623                                <1>       ; (not accessable/meaningful address value by user)
12624 00003E79 AD                    <1>       lodsd
12625 00003E7A A3[528D0100]          <1>       mov    [VGA_INT43H], eax ; VGA current (default) font address
12626                                <1>
12627                                <1>       ; (total 40 bytes are read&written above as BIOS data state)
12628                                <1> bfn_rvs_9:
12629                                <1>       ; 13/01/2021
12630 00003E7F F6C104                <1>       test   cl, 4
12631 00003E82 7422                  <1>       jz     short bfn_rvs_11
12632                                <1>
12633                                <1>       ;BX++;
12634                                <1>         ;v = read_byte(ES, BX); BX++;
12635                                <1>         ;outb(VGAREG_PEL_MASK, read_byte(ES, BX)); BX++;
12636                                <1>         ;// Set the whole dac always, from 0
12637                                <1>         ;outb(VGAREG_DAC_WRITE_ADDRESS,0x00);
12638                                <1>         ;for(i=0;i<256*3;i++) {
12639                                <1>         ;   outb(VGAREG_DAC_DATA, read_byte(ES, BX)); BX++;
12640                                <1>         ;}
12641                                <1>         ;BX++;
12642                                <1>         ;outb(VGAREG_DAC_WRITE_ADDRESS, v);
12643                                <1>
12644                                <1>       ; /* read/write mode dac */
12645 00003E84 AC                    <1>       lodsb  ; skip ; VGAREG_DAC_STATE
12646 00003E85 AC                    <1>       lodsb
12647 00003E86 88C4                  <1>       mov    ah, al ; * ; v
12648 00003E88 AC                    <1>       lodsb
12649 00003E89 66BAC603              <1>       mov    dx, VGAREG_PEL_MASK  ; 3C6h
12650 00003E8D EE                    <1>       out    dx, al
12651                                <1>       ;// Set the whole dac always, from 0
12652 00003E8E 30C0                  <1>       xor    al, al ; 0
```

```
12653                                  <1>       ;mov   dx, VGAREG_DAC_WRITE_ADDRESS ; 3C8h
12654 00003E90 B2C8                    <1>       mov    dl, 0C8h
12655 00003E92 EE                      <1>       out    dx, al
12656                                  <1>
12657 00003E93 51                      <1>       push   ecx ; 22/01/2021
12658                                  <1>       ;for(i=0;i<256*3;i++) {
12659 00003E94 B900030000              <1>       mov    ecx, 256*3 ; 768 bytes
12660                                  <1>       ;mov   dx, VGAREG_DAC_DATA ; 3C9h
12661                                  <1>       ;mov   dl, 0C9h
12662 00003E99 FEC2                    <1>       inc    dl ; dx = 3C9h
12663                                  <1> bfn_rvs_10:
12664 00003E9B AC                      <1>       lodsb
12665 00003E9C EE                      <1>       out    dx, al
12666 00003E9D E2FC                    <1>       loop   bfn_rvs_10
12667 00003E9F 59                      <1>       pop    ecx ; 22/01/2021
12668                                  <1>
12669                                  <1>       ; /* color select register */
12670 00003EA0 AC                      <1>       lodsb   ; skip
12671                                  <1>
12672 00003EA1 88E0                    <1>       mov    al, ah ; * ; v
12673                                  <1>
12674                                  <1>       ;mov   dx, VGAREG_DAC_WRITE_ADDRESS ; 3C8h
12675                                  <1>       ;mov   dl, 0C8h
12676 00003EA3 FECA                    <1>       dec    dl ; dx  = 3C8h
12677 00003EA5 EE                      <1>       out    dx, al ; * ; v
12678                                  <1>
12679                                  <1>       ; (total 772 bytes are read above as DAC state)
12680                                  <1> bfn_rvs_11:
12681 00003EA6 C3                      <1>       retn
12682                                  <1>
12683                                  <1> vbe_biosfn_restore_video_state:
12684                                  <1>       ; 23/01/2021
12685                                  <1>       ; 13/01/2021 (TRDOS 386 v2.0.3)
12686                                  <1>       ; (vbe.c)
12687                                  <1>
12688                                  <1>       ; modified registers: eax, edx, esi, ch
12689                                  <1>
12690                                  <1>       ; input: esi = state buffer address
12691                                  <1>       ; output:
12692                                  <1>       ;      VBE DISPI register contents will be restored
12693                                  <1>       ;      (18 bytes, 9 words)
12694                                  <1>
12695                                  <1>       ; enable = read_word(ES, BX);
12696                                  <1>       ; BX += 2;
12697                                  <1>       ;
12698                                  <1>       ; if (!(enable & VBE_DISPI_ENABLED)) {
12699                                  <1>         ;   outw(VBE_DISPI_IOPORT_INDEX,VBE_DISPI_INDEX_ENABLE);
12700                                  <1>         ;   outw(VBE_DISPI_IOPORT_DATA, enable);
12701                                  <1>       ; } else {
12702                                  <1>         ;   outw(VBE_DISPI_IOPORT_INDEX, VBE_DISPI_INDEX_XRES);
12703                                  <1>         ;   outw(VBE_DISPI_IOPORT_DATA, read_word(ES, BX));
12704                                  <1>         ;   BX += 2;
12705                                  <1>         ;   outw(VBE_DISPI_IOPORT_INDEX, VBE_DISPI_INDEX_YRES);
12706                                  <1>         ;   outw(VBE_DISPI_IOPORT_DATA, read_word(ES, BX));
12707                                  <1>         ;   BX += 2;
12708                                  <1>         ;   outw(VBE_DISPI_IOPORT_INDEX, VBE_DISPI_INDEX_BPP);
12709                                  <1>         ;   outw(VBE_DISPI_IOPORT_DATA, read_word(ES, BX));
12710                                  <1>         ;   BX += 2;
12711                                  <1>         ;   outw(VBE_DISPI_IOPORT_INDEX,VBE_DISPI_INDEX_ENABLE);
12712                                  <1>         ;   outw(VBE_DISPI_IOPORT_DATA, enable);
12713                                  <1>       ;
12714                                  <1>         ;  for(i = VBE_DISPI_INDEX_BANK; i <= VBE_DISPI_INDEX_Y_OFFSET; i++)
12715                                  <1>       ;    {
12716                                  <1>         ;     outw(VBE_DISPI_IOPORT_INDEX, i);
12717                                  <1>         ;     outw(VBE_DISPI_IOPORT_DATA, read_word(ES, BX));
12718                                  <1>         ;     BX += 2;
12719                                  <1>         ;    }
12720                                  <1>         ; }
12721                                  <1>
12722 00003EA7 66AD                    <1>       lodsw  ; enable (status, enabled=1, disabled=0)
12723 00003EA9 66BACE01                <1>       mov    dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
12724                                  <1>       ; 23/01/2021
12725 00003EAD 6683E001                <1>       and    ax, 1 ; VBE_DISPI_ENABLED
12726 00003EB1 750B                    <1>       jnz    short vbe_bfn_rvs_1
12727                                  <1>       ; ax = 0
12728                                  <1>       ; VBE_DISPI_DISABLED
12729                                  <1> vbe_bfn_rvs_0:
12730                                  <1>       ; enable (disable) dispi
12731                                  <1>       ; dx = 01CEh ; VBE_DISPI_IOPORT_INDEX
12732                                  <1>       ; ah = 0
12733 00003EB3 50                      <1>       push   eax
12734 00003EB4 B004                    <1>       mov    al, 04h      ; VBE_DISPI_INDEX_ENABLE
12735 00003EB6 66EF                    <1>       out    dx, ax
12736                                  <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
12737 00003EB8 FEC2                    <1>       inc    dl
12738 00003EBA 58                      <1>       pop    eax
12739 00003EBB 66EF                    <1>       out    dx, ax ; enable (or disable)
12740 00003EBD C3                      <1>       retn
12741                                  <1> vbe_bfn_rvs_1:
12742                                  <1>       ; VBE_DISPI_ENABLED
12743                                  <1>
12744                                  <1>       ; from VBE_DISPI_INDEX_XRES
12745                                  <1>       ;   to VBE_DISPI_INDEX_BPP
12746                                  <1>
12747 00003EBE B503                    <1>       mov    ch, 3
12748 00003EC0 28C0                    <1>       sub    al, al ; 0 ; VBE_DISPI_INDEX_XRES - 1
12749                                  <1>       ; ax = 0
12750                                  <1>
12751 00003EC2 E80B000000              <1>       call   vbe_bfn_rvs_2
12752                                  <1>
12753                                  <1>         ;outw(VBE_DISPI_IOPORT_INDEX,VBE_DISPI_INDEX_ENABLE);
12754                                  <1>       ;outw(VBE_DISPI_IOPORT_DATA, enable);
12755                                  <1>
12756                                  <1>       ; 23/01/2021
12757 00003EC7 B001                    <1>       mov    al, 1 ; VBE_DISPI_ENABLED
```

```
12758                                  <1>       ; ax = 1
12759 00003EC9 E8E5FFFFFF              <1>       call   vbe_bfn_rvs_0
12760                                  <1>
12761                                  <1>       ; from VBE_DISPI_INDEX_BANK
12762                                  <1>       ;   to VBE_DISPI_INDEX_Y_OFFSET
12763                                  <1>
12764 00003ECE B505                    <1>       mov    ch, 5
12765                                  <1>       ; 23/01/2021
12766 00003ED0 B004                    <1>       mov    al, 4  ; VBE_DISPI_INDEX_BANK - 1
12767                                  <1>       ; ax = 4
12768                                  <1> vbe_bfn_rvs_2:
12769 00003ED2 FEC0                    <1>       inc    al ; from VBE_DISPI_INDEX_XRES
12770                                  <1>                     ; to VBE_DISPI_INDEX_BPP
12771                                  <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
12772                                  <1>       ;mov   dl, 0CEh
12773 00003ED4 66EF                   <1>       out    dx, ax
12774 00003ED6 50                      <1>       push   eax
12775                                  <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
12776 00003ED7 FEC2                    <1>       inc    dl ; 1CFh
12777 00003ED9 66AD                    <1>       lodsw
12778 00003EDB 66EF                    <1>       out    dx, ax
12779 00003EDD 58                      <1>       pop    eax
12780 00003EDE FECA                    <1>       dec    dl ; 1CEh
12781 00003EE0 FECD                    <1>       dec    ch
12782 00003EE2 75EE                    <1>       jnz    short vbe_bfn_rvs_2
12783 00003EE4 C3                      <1>       retn
12784                                  <1>
12785                                  <1> ; ----------------------------------------------------------
12786                                  <1>
12787                                  <1> dispi_set_enable:
12788                                  <1>       ; 23/11/2020
12789                                  <1>       ; Input:
12790                                  <1>       ;      ax = VBE_DISPI_ENABLED = 1
12791                                  <1>       ;           or VBE_DISPI_DISABLED = 0
12792                                  <1>       ;
12793                                  <1>       ; Modified registers: none
12794                                  <1>
12795                                  <1>       ;push  edx
12796                                  <1>       ;push  eax
12797                                  <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
12798                                  <1>       ;mov   ax, 04h         ; VBE_DISPI_INDEX_ENABLE
12799                                  <1>       ;out   dx, ax
12800                                  <1>       ;pop   eax
12801                                  <1>       ;;mov  dx, 01CFh ; VBE_DISPI_IOPORT_DATA
12802                                  <1>       ;;mov  dl, 0CFh
12803                                  <1>       ;inc   dl
12804                                  <1>       ;out   dx, ax
12805                                  <1>       ;pop   edx
12806                                  <1>       ;retn
12807                                  <1>
12808                                  <1>       ; 25/11/2020
12809                                  <1>       ; Modified registers: edx
12810                                  <1>       ;push  edx
12811 00003EE5 66BA0400                <1>       mov    dx, 04h ; VBE_DISPI_INDEX_ENABLE
12812                                  <1>       ;call  dispi_set_parms
12813                                  <1>       ;pop   edx
12814                                  <1>       ;retn
12815                                  <1>       ;;jmp  short dispi_set_parms
12816                                  <1>
12817                                  <1> dispi_set_parms:
12818                                  <1>       ; 25/11/2020
12819                                  <1>       ; Input:
12820                                  <1>       ;      ax = data
12821                                  <1>       ;      dx = vbe dispi register index
12822                                  <1>       ;
12823                                  <1>       ; Modified registers: edx
12824                                  <1>
12825 00003EE9 50                      <1>       push   eax
12826 00003EEA 6689D0                  <1>       mov    ax, dx
12827 00003EED 66BACE01                <1>       mov    dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
12828 00003EF1 66EF                    <1>       out    dx, ax
12829 00003EF3 58                      <1>       pop    eax
12830                                  <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
12831                                  <1>       ;mov   dl, 0CFh
12832 00003EF4 FEC2                    <1>       inc    dl
12833 00003EF6 66EF                    <1>       out    dx, ax
12834 00003EF8 C3                      <1>       retn
12835                                  <1>
12836                                  <1> dispi_set_bpp:
12837                                  <1>       ; 25/11/2020
12838                                  <1>       ; Input:
12839                                  <1>       ;      ax = Bits per pixel value
12840                                  <1>       ;           (8,16,24,32)
12841                                  <1>       ;
12842                                  <1>       ; Modified registers: none
12843                                  <1>
12844                                  <1>       ;push  edx
12845                                  <1>       ;push  eax
12846                                  <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
12847                                  <1>       ;mov   ax, 03h         ; VBE_DISPI_INDEX_BPP
12848                                  <1>       ;out   dx, ax
12849                                  <1>       ;pop   eax
12850                                  <1>       ;;mov  dx, 01CFh ; VBE_DISPI_IOPORT_DATA
12851                                  <1>       ;;mov  dl, 0CFh
12852                                  <1>       ;inc   dl
12853                                  <1>       ;out   dx, ax
12854                                  <1>       ;pop   edx
12855                                  <1>       ;retn
12856                                  <1>
12857                                  <1>       ; 25/11/2020
12858                                  <1>       ; Modified registers: edx
12859                                  <1>       ;push  edx
12860 00003EF9 66BA0300                <1>       mov    dx, 03h ; VBE_DISPI_INDEX_BPP
12861                                  <1>       ;call  dispi_set_parms
12862                                  <1>       ;pop   edx
```

```
12863                                  <1>       ;retn
12864 00003EFD EBEA                    <1>       jmp    short dispi_set_parms
12865                                  <1>
12866                                  <1> dispi_set_xres:
12867                                  <1>       ; 25/11/2020
12868                                  <1>       ; Input:
12869                                  <1>       ;      ax = X resolution (screen witdh)
12870                                  <1>       ;           (320,640,800,1024,1280,1920)
12871                                  <1>       ;
12872                                  <1>       ; Modified registers: none
12873                                  <1>
12874                                  <1>       ;push  edx
12875                                  <1>       ;push  eax
12876                                  <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
12877                                  <1>       ;mov   ax, 01h        ; VBE_DISPI_INDEX_XRES
12878                                  <1>       ;out   dx, ax
12879                                  <1>       ;pop   eax
12880                                  <1>       ;;mov  dx, 01CFh ; VBE_DISPI_IOPORT_DATA
12881                                  <1>       ;;mov  dl, 0CFh
12882                                  <1>       ;inc   dl
12883                                  <1>       ;out   dx, ax
12884                                  <1>       ;pop   edx
12885                                  <1>       ;retn
12886                                  <1>
12887                                  <1>       ; 25/11/2020
12888                                  <1>       ; Modified registers: edx
12889                                  <1>       ;push  edx
12890 00003EFF 66BA0100                <1>       mov    dx, 01h ; VBE_DISPI_INDEX_XRES
12891                                  <1>       ;call  dispi_set_parms
12892                                  <1>       ;pop   edx
12893                                  <1>       ;retn
12894 00003F03 EBE4                    <1>       jmp    short dispi_set_parms
12895                                  <1>
12896                                  <1> dispi_set_yres:
12897                                  <1>       ; 25/11/2020
12898                                  <1>       ; Input:
12899                                  <1>       ;      ax = Y resolution (screen height)
12900                                  <1>       ;           (200,400,600,720,768,1080)
12901                                  <1>       ;
12902                                  <1>       ; Modified registers: none
12903                                  <1>
12904                                  <1>       ;push  edx
12905                                  <1>       ;push  eax
12906                                  <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
12907                                  <1>       ;mov   ax, 02h        ; VBE_DISPI_INDEX_YRES
12908                                  <1>       ;out   dx, ax
12909                                  <1>       ;pop   eax
12910                                  <1>       ;;mov  dx, 01CFh ; VBE_DISPI_IOPORT_DATA
12911                                  <1>       ;;mov  dl, 0CFh
12912                                  <1>       ;inc   dl
12913                                  <1>       ;out   dx, ax
12914                                  <1>       ;pop   edx
12915                                  <1>       ;retn
12916                                  <1>
12917                                  <1>       ; 25/11/2020
12918                                  <1>       ; Modified registers: edx
12919                                  <1>       ;push  edx
12920 00003F05 66BA0200                <1>       mov    dx, 02h  ; VBE_DISPI_INDEX_YRES
12921                                  <1>       ;call  dispi_set_parms
12922                                  <1>       ;pop   edx
12923                                  <1>       ;retn
12924 00003F09 EBDE                    <1>       jmp    short dispi_set_parms
12925                                  <1>
12926                                  <1> dispi_set_bank:
12927                                  <1>       ; 25/11/2020
12928                                  <1>       ; Input:
12929                                  <1>       ;      ax = video memory bank number
12930                                  <1>       ;
12931                                  <1>       ; Modified registers: none
12932                                  <1>
12933                                  <1>       ;push  edx
12934                                  <1>       ;push  eax
12935                                  <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
12936                                  <1>       ;mov   ax, 05h        ; VBE_DISPI_INDEX_BANK
12937                                  <1>       ;out   dx, ax
12938                                  <1>       ;pop   eax
12939                                  <1>       ;;mov  dx, 01CFh ; VBE_DISPI_IOPORT_DATA
12940                                  <1>       ;;mov  dl, 0CFh
12941                                  <1>       ;inc   dl
12942                                  <1>       ;out   dx, ax
12943                                  <1>       ;pop   edx
12944                                  <1>       ;retn
12945                                  <1>
12946                                  <1>       ; 25/11/2020
12947                                  <1>       ; Modified registers: edx
12948                                  <1>       ;push  edx
12949 00003F0B 66BA0500                <1>       mov    dx, 05h  ; VBE_DISPI_INDEX_BANK
12950                                  <1>       ;call  dispi_set_parms
12951                                  <1>       ;pop   edx
12952                                  <1>       ;retn
12953 00003F0F EBD8                    <1>       jmp    short dispi_set_parms
12954                                  <1>
12955                                  <1> dispi_get_enable:
12956                                  <1>       ; 27/11/2020
12957                                  <1>       ; Input:
12958                                  <1>       ;      none
12959                                  <1>       ; Output:
12960                                  <1>       ;      ax = vbe dispi status
12961                                  <1>       ;
12962                                  <1>       ; Modified registers: eax
12963                                  <1>
12964                                  <1>       ;push  edx
12965                                  <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
12966                                  <1>       ;mov   ax, 04h        ; VBE_DISPI_INDEX_ENABLE
12967                                  <1>       ;out   dx, ax
```

```
                        <1>       ;;mov  dx, 01CFh ; VBE_DISPI_IOPORT_DATA
                        <1>       ;;mov  dl, 0CFh
                        <1>       ;inc   dl
                        <1>       ;in    ax, dx
                        <1>       ;pop   edx
                        <1>       ;retn
                        <1>
                        <1>       ; 27/11/2020
                        <1>       ; Modified registers: eax, edx
                        <1>       ;push  edx
00003F11 66B80400       <1>       mov    ax, 04h ; VBE_DISPI_INDEX_ENABLE
                        <1>       ;call  dispi_get_parms
                        <1>       ;pop   edx
                        <1>       ;retn
                        <1>       ;;jmp  short dispi_get_parms
                        <1>
                        <1> dispi_get_parms:
                        <1>       ; 25/11/2020
                        <1>       ; Input:
                        <1>       ;      ax = vbe dispi register index
                        <1>       ; output:
                        <1>       ;      ax = data
                        <1>       ;
                        <1>       ; Modified registers: eax, edx
                        <1>
00003F15 66BACE01       <1>       mov    dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
00003F19 66EF           <1>       out    dx, ax
                        <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
                        <1>       ;mov   dl, 0CFh
00003F1B FEC2           <1>       inc    dl
00003F1D 66ED           <1>       in     ax, dx
00003F1F C3             <1>       retn
                        <1>
                        <1> vga_compat_setup:
                        <1>       ; 26/11/2020
                        <1>       ; 25/11/2020
                        <1>       ; VGA compatibility setup
                        <1>       ; (vbe.c, 02/01/2020, vruppert)
                        <1>       ;
                        <1>       ; Input:
                        <1>       ;      none
                        <1>       ;
                        <1>       ; Modified registers: eax, edx
                        <1>
                        <1>       ; 26/11/2020
                        <1>       ;push  eax
                        <1>       ;push  edx
                        <1>
                        <1>       ; set CRT X resolution
00003F20 66BACE01       <1>       mov    dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
00003F24 66B80100       <1>       mov    ax, 01h  ; VBE_DISPI_INDEX_XRES
00003F28 66EF           <1>       out    dx, ax
                        <1>       ;mov   dx, 1CFh ; VBE_DISPI_IOPORT_DATA
00003F2A FEC2           <1>       inc    dl
00003F2C 66ED           <1>       in     ax, dx
00003F2E 50             <1>       push   eax
00003F2F 66BAD403       <1>       mov    dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
00003F33 66B81100       <1>       mov    ax, 0011h ; Vertical retrace end register
00003F37 66EF           <1>       out    dx, ax
                        <1>       ;pop   eax
                        <1>       ;push  eax
00003F39 8B0424         <1>       mov    eax, [esp]
00003F3C 66C1E803       <1>       shr    ax, 3 ; / 8 for pixel to character
00003F40 6648           <1>       dec    ax  ; - 1 (EGA or VGA?)
00003F42 88C4           <1>       mov    ah, al
00003F44 B001           <1>       mov    al, 01h ; Horizontal display end register
00003F46 66EF           <1>       out    dx, ax
00003F48 58             <1>       pop    eax
                        <1>
00003F49 E8B0000000     <1>       call   vga_set_virt_width
                        <1>
                        <1>       ; set CRT Y resolution
00003F4E 66BACE01       <1>       mov    dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
00003F52 66B80200       <1>       mov    ax, 02h  ; VBE_DISPI_INDEX_YRES
00003F56 66EF           <1>       out    dx, ax
                        <1>       ;mov   dx, 1CFh ; VBE_DISPI_IOPORT_DATA
00003F58 FEC2           <1>       inc    dl
00003F5A 66ED           <1>       in     ax, dx
00003F5C 50             <1>       push   eax
00003F5D 66BAD403       <1>       mov    dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
00003F61 88C4           <1>       mov    ah, al
00003F63 B012           <1>       mov    al, 12h ; Vertical display end register
00003F65 66EF           <1>       out    dx, ax
00003F67 58             <1>       pop    eax
00003F68 B007           <1>       mov    al, 07h      ; Overflow register
00003F6A EE             <1>       out    dx, al
00003F6B 6642           <1>       inc    dx
00003F6D EC             <1>       in     al, dx ; read overflow register
00003F6E 24BD           <1>       and    al, 0BDh ; clear VDE 9th and 10th bits
00003F70 F6C401         <1>       test   ah, 01h
00003F73 7402           <1>       jz     short bit8_clear
00003F75 0C02           <1>       or     al, 02h ; VDE 9th bit (bit 8) in bit 1
                        <1> bit8_clear:
00003F77 F6C402         <1>       test   ah, 02h
00003F7A 7402           <1>       jz     short bit9_clear
00003F7C 0C40           <1>       or     al, 40h ; VDE 10th bit (bit 9) in bit 6
                        <1> bit9_clear:
00003F7E EE             <1>       out    dx, al
                        <1>
                        <1>       ; other settings
00003F7F 66BAD403       <1>       mov    dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
00003F83 66B80900       <1>       mov    ax, 0009h ; Maximum scan line register
00003F87 66EF           <1>       out    dx, ax ; Reset
00003F89 B017           <1>       mov    al, 17h ; Mode control register
00003F8B EE             <1>       out    dx, al
```

```
13073                                  <1>       ;mov   dx, 3D5h ; VGAREG_VGA_CRTC_DATA
13074 00003F8C FEC2                    <1>       inc    dl
13075 00003F8E EC                      <1>       in     al, dx ; Read mode control register
13076 00003F8F 0C03                    <1>       or     al, 03h ; Set SRS and CMS bits
13077 00003F91 EE                      <1>       out    dx, al
13078 00003F92 66BADA03                <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
13079 00003F96 EC                      <1>       in     al, dx  ; clear flip-flop
13080 00003F97 66BAC003                <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
13081 00003F9B B010                    <1>       mov    al, 10h      ; Mode control register
13082 00003F9D EE                      <1>       out    dx, al
13083                                  <1>       ;mov   dx, 3C1h ; VGAREG_ACTL_READ_DATA
13084 00003F9E FEC2                    <1>       inc    dl
13085 00003FA0 EC                      <1>       in     al, dx
13086 00003FA1 0C01                    <1>       or     al, 01h ; select graphics mode
13087                                  <1>       ;mov   dx, 3C0h ; VGAREG_ACTL_ADDRESS
13088 00003FA3 FECA                    <1>       dec    dl
13089 00003FA5 EE                      <1>       out    dx, al ; Write to mode control register
13090 00003FA6 B020                    <1>       mov    al, 20h ; Palette RAM <-> display memory
13091 00003FA8 EE                      <1>       out    dx, al ; Write to attribute addr register
13092 00003FA9 66BACE03                <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
13093 00003FAD 66B80605                <1>       mov    ax, 0506h ; Misc. register, graph, mm 1
13094 00003FB1 66EF                    <1>       out    dx, ax
13095 00003FB3 66BAC403                <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
13096 00003FB7 66B8020F                <1>       mov    ax, 0F02h ; Map mask register, all planes
13097 00003FBB 66EF                    <1>       out    dx, ax
13098                                  <1>
13099                                  <1>       ; settings for >= 8bpp
13100                                  <1>
13101                                  <1>       ;mov   dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
13102                                  <1>       ;mov   ax, 03h ; VBE_DISPI_INDEX_BPP
13103                                  <1>       ;out   dx, ax
13104                                  <1>       ;;mov  dx, 1CFh ; VBE_DISPI_IOPORT_DATA
13105                                  <1>       ;inc   dl
13106                                  <1>       ;in    ax, dx
13107                                  <1>       ;cmp   al, 08h  ; < 8 bits per pixel
13108                                  <1>       ;jb    short vga_compat_end
13109                                  <1>
13110 00003FBD 66BAD403                <1>       mov    dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
13111 00003FC1 B014                    <1>       mov    al, 14h  ; Underline location register
13112 00003FC3 EE                      <1>       out    dx, al
13113                                  <1>       ;mov   dx, 3D5h ; VGAREG_VGA_CRTC_DATA
13114 00003FC4 FEC2                    <1>       inc    dl
13115 00003FC6 EC                      <1>       in     al, dx
13116 00003FC7 0C40                    <1>       or     al, 40h       ; enable double word mode
13117 00003FC9 EE                      <1>       out    dx, al
13118 00003FCA 66BADA03                <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
13119 00003FCE EC                      <1>       in     al, dx  ; clear flip-flop
13120 00003FCF 66BAC003                <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
13121 00003FD3 B010                    <1>       mov    al, 10h       ; Mode control register
13122 00003FD5 EE                      <1>       out    dx, al
13123                                  <1>       ;mov   dx, 3C1h ; VGAREG_ACTL_READ_DATA
13124 00003FD6 FEC2                    <1>       inc    dl
13125 00003FD8 EC                      <1>       in     al, dx
13126 00003FD9 0C40                    <1>       or     al, 40h  ; Pixel clock select is 1
13127                                  <1>       ;mov   dx, 3C0h ; VGAREG_ACTL_ADDRESS
13128 00003FDB FECA                    <1>       dec    dl
13129 00003FDD EE                      <1>       out    dx, al  ; update mode control reggister
13130 00003FDE B020                    <1>       mov    al, 20h       ; select display memory as PAS
13131 00003FE0 EE                      <1>       out    dx, al
13132 00003FE1 66BAC403                <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
13133 00003FE5 B004                    <1>       mov    al, 04h      ; Memory mode register
13134 00003FE7 EE                      <1>       out    dx, al
13135                                  <1>       ;mov   dx, 3C5h ; VGAREG_SEQU_DATA
13136 00003FE8 FEC2                    <1>       inc    dl
13137 00003FEA EC                      <1>       in     al, dx
13138 00003FEB 0C08                    <1>       or     al, 08h      ; enable chain four
13139 00003FED EE                      <1>       out    dx, al
13140 00003FEE 66BACE03                <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
13141 00003FF2 B005                    <1>       mov    al, 05h       ; Mode register
13142 00003FF4 EE                      <1>       out    dx, al
13143                                  <1>       ;mov   dx, 3CFh ; VGAREG_GRDC_DATA
13144 00003FF5 FEC2                    <1>       inc    dl
13145 00003FF7 EC                      <1>       in     al, dx
13146 00003FF8 249F                    <1>       and    al, 9Fh       ; clear shift register
13147 00003FFA 0C40                    <1>       or     al, 40h  ; set shift register to 2
13148 00003FFC EE                      <1>       out    dx, al
13149                                  <1>
13150                                  <1> vga_compat_end:
13151                                  <1>       ;pop   edx
13152                                  <1>       ;pop   eax
13153 00003FFD C3                      <1>       retn
13154                                  <1>
13155                                  <1> vga_set_virt_width:
13156                                  <1>       ; 27/11/2020
13157                                  <1>       ; 25/11/2020
13158                                  <1>       ; (vbe.c, 02/01/2020, vruppert)
13159                                  <1>       ;
13160                                  <1>       ; Input:
13161                                  <1>       ;      AX = resolution (screen width)
13162                                  <1>       ;
13163                                  <1>       ; Modified registers: eax, edx
13164                                  <1>
13165                                  <1>       ;;push ebx
13166                                  <1>       ;push  edx
13167                                  <1>       ;push  eax
13168                                  <1>       ;mov   ebx, eax
13169                                  <1>       ;call  dispi_get_bpp ; bits per pixel
13170                                  <1>       ;cmp   al, 4
13171                                  <1>       ;ja    short set_width_svga  ; 8, 16, 24, 32
13172                                  <1>       ;shr   bx, 1
13173                                  <1> ;set_width_svga:
13174                                  <1>       ;shr   bx, 3
13175                                  <1>       ;mov   eax, [esp]
13176 00003FFE 66C1E803               <1>       shr    ax, 3  ; / 8, bytes per row
13177 00004002 66BAD403                <1>       mov    dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
```

```
13178                                   <1>       ;mov   ah, bl ;
13179 00004006 88C4                     <1>       mov    ah, al ; width in bytes
13180 00004008 B013                     <1>       mov    al, 13h      ; offset register
13181 0000400A 66EF                     <1>       out    dx, ax ; index (3D4h) and data (3D5h)
13182                                   <1>       ;pop   eax
13183                                   <1>       ;pop   edx
13184                                   <1>       ;;pop  ebx
13185 0000400C C3                       <1>       retn
13186                                   <1>
13187                                   <1> ; 24/11/2020
13188                                   <1>
13189                                   <1> struc bmi ; BOCHS/PLEX86 MODE INFO structure/table
13190 00000000 ????                     <1>  .mode:        resw 1
13191 00000002 ????                     <1>  .width:  resw 1
13192 00000004 ????                     <1>  .height: resw 1
13193 00000006 ????                     <1>  .depth:  resw 1
13194                                   <1>  .size:
13195                                   <1> endstruc
13196                                   <1>
13197                                   <1> ; 24/11/2020
13198                                   <1> struc MODEINFO
13199 00000000 ????                     <1>  .mode:                    resw 1  ; 1XXh
13200 00000002 ????                     <1>  .ModeAttributes:  resw 1
13201 00000004 ??                       <1>  .WinAAttributes:  resb 1
13202 00000005 ??                       <1>  .WinBAttributes:  resb 1 ; = 0
13203 00000006 ????                     <1>  .WinGranularity:  resw 1
13204 00000008 ????                     <1>  .WinSize:         resw 1
13205 0000000A ????                     <1>  .WinASegment:             resw 1
13206 0000000C ????                     <1>  .WinBSegment:             resw 1 ; = 0
13207 0000000E ????????                 <1>  .WinFuncPtr:              resd 1 ; = 0
13208 00000012 ????                     <1>  .BytesPerScanLine:        resw 1
13209 00000014 ????                     <1>  .XResolution:             resw 1
13210 00000016 ????                     <1>  .YResolution:             resw 1
13211 00000018 ??                       <1>  .XCharSize:       resb 1
13212 00000019 ??                       <1>  .YCharSize:       resb 1
13213 0000001A ??                       <1>  .NumberOfPlanes:  resb 1
13214 0000001B ??                       <1>  .BitsPerPixel:            resb 1
13215 0000001C ??                       <1>  .NumberOfBanks:   resb 1
13216 0000001D ??                       <1>  .MemoryModel:             resb 1
13217 0000001E ??                       <1>  .BankSize:        resb 1 ; = 0
13218 0000001F ??                       <1>  .NumberOfImagePages:      resb 1
13219 00000020 ??                       <1>  .Reserved_page:   resb 1 ; = 0
13220 00000021 ??                       <1>  .RedMaskSize:             resb 1
13221 00000022 ??                       <1>  .RedFieldPosition:        resb 1
13222 00000023 ??                       <1>  .GreenMaskSize:   resb 1
13223 00000024 ??                       <1>  .GreenFieldPosition:      resb 1
13224 00000025 ??                       <1>  .BlueMaskSize:            resb 1
13225 00000026 ??                       <1>  .BlueFieldPosition:       resb 1
13226 00000027 ??                       <1>  .RsvdMaskSize:            resb 1
13227 00000028 ??                       <1>  .RsvdFieldPosition:       resb 1
13228 00000029 ??                       <1>  .DirectColorModeInfo:     resb 1
13229 0000002A ????????                 <1>  .PhysBasePtr:             resd 1
13230 0000002E ????????                 <1>  .OffScreenMemOffset:      resd 1 ; = 0
13231 00000032 ????                     <1>  .OffScreenMemSize:        resw 1 ; = 0
13232 00000034 ????                     <1>  .LinBytesPerScanLine:     resw 1
13233 00000036 ??                       <1>  .BnkNumberOfPages:        resb 1
13234 00000037 ??                       <1>  .LinNumberOfPages:        resb 1
13235 00000038 ??                       <1>  .LinRedMaskSize:  resb 1
13236 00000039 ??                       <1>  .LinRedFieldPosition1: resb 1
13237 0000003A ??                       <1>  .LinGreenMaskSize1:       resb 1
13238 0000003B ??                       <1>  .LinGreenFieldPosition:resb 1
13239 0000003C ??                       <1>  .LinBlueMaskSize:         resb 1
13240 0000003D ??                       <1>  .LinBlueFieldPosition:    resb 1
13241 0000003E ??                       <1>  .LinRsvdMaskSize:         resb 1
13242 0000003F ??                       <1>  .LinRsvdFieldPosition:    resb 1
13243 00000040 ????????                 <1>  .MaxPixelClock:   resd 1 ; = 0
13244                                   <1>  .size:
13245                                   <1> endstruc
13246                                   <1>
13247                                   <1> ; 10/12/2020
13248                                   <1> struc LFBINFO
13249 00000000 ????                     <1>  .mode:                    resw 1  ; 1XXh
13250 00000002 ????????                 <1>  .LFB_addr:        resd 1
13251 00000006 ????????                 <1>  .LFB_size:        resd 1
13252 0000000A ????                     <1>  .X_res:           resw 1
13253 0000000C ????                     <1>  .Y_res:           resw 1
13254 0000000E ??                       <1>  .bpp:                     resb 1
13255 0000000F ??                       <1>  .reserved:        resb 1
13256                                   <1> .size:      ; 16 bytes
13257                                   <1> endstruc
13258                                   <1>
13259                                   <1> set_mode_info_list:
13260                                   <1>       ; 14/12/2020
13261                                   <1>       ; 11/12/2020
13262                                   <1>       ; 24/11/2020
13263                                   <1>       ; (vbetables-gen.c)
13264                                   <1>       ; Input:
13265                                   <1>       ;      BX = VBE mode (including bochs special modes)
13266                                   <1>       ; Output:
13267                                   <1>       ;      ;;EAX = MODE_INFO_LIST address
13268                                   <1>       ;      EAX = 0 ; 11/12/2020
13269                                   <1>       ;      ESI = MODE_INFO_LIST address ; 11/12/2020
13270                                   <1>       ;      (if mode is not found, ESI = 0)
13271                                   <1>       ;
13272                                   <1>       ; Modified registers: eax, ebx, ecx, edx, esi, edi
13273                                   <1>
13274 0000400D BE[466E0000]             <1>       mov    esi, b_vbe_modes ; bochs mode info base table
13275 00004012 BF[22120300]             <1>       mov    edi, MODE_INFO_LIST ; mode info list (4F01h)
13276                                   <1> sml_0:
13277 00004017 66AD                     <1>       lodsw
13278 00004019 6639D8                   <1>       cmp    ax, bx ; is mode number same ?
13279 0000401C 7410                     <1>       je     short sml_1 ; yes
13280 0000401E AD                       <1>       lodsd
13281 0000401F 66AD                     <1>       lodsw
13282 00004021 81FE[066F0000]           <1>       cmp    esi, end_of_b_vbe_modes
```

```
13283 00004027 72EE              <1>       jb     short sml_0
13284                            <1>       ; not found
13285 00004029 31C0              <1>       xor    eax, eax ; 0
13286                            <1>       ; 11/12/2020
13287 0000402B 31F6              <1>       xor    esi, esi
13288 0000402D C3                <1>       retn
13289                            <1> sml_1:
13290 0000402E 66AB              <1>       stosw  ; mode
13291 00004030 AD                <1>       lodsd  ; width, height
13292                            <1>       ; 14/12/2020
13293 00004031 89C1              <1>       mov    ecx, eax
13294 00004033 50                <1>       push   eax ; ***
13295 00004034 29C0              <1>       sub    eax, eax ; clear high word of eax
13296 00004036 66AD              <1>       lodsw  ; depth
13297 00004038 50                <1>       push   eax ; **
13298                            <1>
13299                            <1>       ;add   al, 7 ; only for 15 bit colors (not used here)
13300 00004039 C0E803            <1>       shr    al, 3 ; / 8
13301                            <1>       ; 14/12/2020
13302 0000403C 66F7E1            <1>       mul    cx  ; pitch = width * ((depth+7)/8)
13303                            <1>              ; ax = pitch
13304 0000403F 50                <1>       push   eax ; * ; high word of eax = 0
13305 00004040 C1E910            <1>       shr    ecx, 16
13306                            <1>       ;mul   cx
13307                            <1>       ;mov   cx, ax
13308 00004043 31D2              <1>       xor    edx, edx  ; clear high word of edx
13309 00004045 F7E1              <1>       mul    ecx ; height * pitch
13310 00004047 89C1              <1>       mov    ecx, eax
13311 00004049 B800000001        <1>       mov    eax, VBE_DISPI_TOTAL_VIDEO_MEMORY_MB * 1024 * 1024
13312 0000404E F7F1              <1>       div    ecx
13313                            <1>              ; eax = pages = vram_size / (height*pitch)
13314                            <1>
13315                            <1>       ;mov   cx, ax
13316 00004050 89C1              <1>       mov    ecx, eax ; pages
13317                            <1>
13318 00004052 66B89B00          <1>       mov    ax, MODE_ATTRIBUTES
13319 00004056 66AB              <1>       stosw  ; ModeAttributes
13320 00004058 B007              <1>       mov    al, WINA_ATTRIBUTES
13321 0000405A AA                <1>       stosb  ; WinAAttributes
13322 0000405B 30C0              <1>       xor    al, al ; WinBAttributes = 0
13323 0000405D AA                <1>       stosb
13324 0000405E 66B84000          <1>       mov    ax, VBE_DISPI_BANK_SIZE_KB
13325 00004062 66AB              <1>       stosw  ; WinGranularity
13326 00004064 66AB              <1>       stosw  ; WinSize
13327 00004066 66B800A0          <1>       mov    ax, VGAMEM_GRAPH
13328 0000406A 66AB              <1>       stosw  ; WinASegment
13329 0000406C 29C0              <1>       sub    eax, eax
13330 0000406E 66AB              <1>       stosw  ; WinBSegment = 0
13331 00004070 AB                <1>       stosd  ; WinFuncPtr = 0
13332                            <1>
13333 00004071 58                <1>       pop    eax ; * ; pitch
13334 00004072 89C3              <1>       mov    ebx, eax  ; high word of ebx = 0 ; 14/12/2020
13335 00004074 66AB              <1>       stosw  ; BytesPerScanLine
13336                            <1>
13337 00004076 5A                <1>       pop    edx ; ** ; depth (bits per pixel)
13338 00004077 58                <1>       pop    eax ; *** width, height
13339                            <1>
13340                            <1>       ; // Mandatory information for VBE 1.2 and above
13341                            <1>
13342 00004078 66AB              <1>       stosw  ; XResolution (width)
13343 0000407A C1E810            <1>       shr    eax, 16
13344 0000407D 50                <1>       push   eax ; **** height
13345 0000407E 66AB              <1>       stosw  ; YResolution (height)
13346 00004080 B008              <1>       mov    al, 8
13347 00004082 AA                <1>       stosb  ; XCharSize  ; char width
13348 00004083 B010              <1>       mov    al, 16
13349 00004085 AA                <1>       stosb  ; YCharSize  ; char height
13350 00004086 B001              <1>       mov    al, 1
13351 00004088 AA                <1>       stosb  ; NumberOfPlanes
13352                            <1>       ;movzx eax, dl
13353 00004089 88D0              <1>       mov    al, dl ; eax <= 32
13354 0000408B AA                <1>       stosb  ; BitsPerPixel
13355                            <1>       ; Number of banks = (height * pitch + 65535) / 65536
13356 0000408C 58                <1>       pop    eax ; **** ; height
13357                            <1>       ; 14/12/2020
13358 0000408D 52                <1>       push   edx ; ***** ; depth ; edx <= 32
13359 0000408E F7E3              <1>       mul    ebx  ; pitch (ebx) * height (eax)
13360                            <1>       ;mov   edx, [esp] ; *****
13361                            <1>       ;mov   dl, [esp] ; *****
13362 00004090 05FFFF0000        <1>       add    eax, 65535
13363 00004095 C1E810            <1>       shr    eax, 16 ; / 65536 ; <= 127 ; 14/12/2020
13364 00004098 AA                <1>       stosb  ; NumberOfBanks
13365                            <1>       ; 14/12/2020
13366                            <1>       ;cmp   dl, 8 ; 8 bits per pixel
13367 00004099 803C2408          <1>       cmp    byte [esp], 8
13368 0000409D 7704              <1>       ja     short sml_2
13369 0000409F B004              <1>       mov    al, VBE_MEMORYMODEL_PACKED_PIXEL
13370 000040A1 EB02              <1>       jmp    short sml_3
13371                            <1> sml_2:
13372                            <1>       ; 16, 24, 32 bits per pixel
13373 000040A3 B006              <1>       mov    al, VBE_MEMORYMODEL_DIRECT_COLOR
13374                            <1> sml_3:
13375 000040A5 AA                <1>       stosb
13376 000040A6 30C0              <1>       xor    al, al ; 0
13377 000040A8 AA                <1>       stosb  ; BankSize = 0
13378 000040A9 49                <1>       dec    ecx ; pages - 1
13379                            <1>       ; NumberOfImagePages = 262 for 320x200x8 mode
13380                            <1>       ;;mov  ax, 255
13381                            <1>       ; 14/12/2020
13382                            <1>       ;mov   al, 255
13383 000040AA FEC8              <1>       dec    al ; 255
13384 000040AC 39C1              <1>       cmp    ecx, eax ; ecx <= 261, eax = 255
13385                            <1>       ;cmp   cx, ax
13386 000040AE 7302              <1>       jnb    short sml_4
13387 000040B0 88C8              <1>       mov    al, cl
```

```
13388                                   <1> sml_4:
13389 000040B2 AA                       <1>         stosb  ; NumberOfImagePages (1 byte)
13390 000040B3 28C0                     <1>         sub    al, al
13391 000040B5 AA                       <1>         stosb  ; Reserved_page = 0
13392 000040B6 58                       <1>         pop    eax ; ***** ; depth
13393 000040B7 88C1                     <1>         mov    cl, al
13394                                   <1>         ; eax < = 32
13395 000040B9 2C08                     <1>         sub    al, 8 ; 8->0, 16->8, 24->16, 32->24
13396 000040BB BE[066F0000]             <1>         mov    esi, direct_color_fields
13397 000040C0 01C6                     <1>         add    esi, eax
13398 000040C2 56                       <1>         push   esi ; ******
13399 000040C3 AD                       <1>         lodsd  ; RedMaskSize (AL), RedFieldPosition (AH)
13400                                   <1>                ; GreenMaskSize (16), GreenFieldPosition (24)
13401 000040C4 AB                       <1>         stosd
13402 000040C5 AD                       <1>         lodsd  ; BlueMaskSize (AL), BlueFieldPosition (AH)
13403                                   <1>                ; RsvdMaskSize (16), RsvdFieldPosition (24)
13404 000040C6 AB                       <1>         stosd
13405 000040C7 5E                       <1>         pop    esi ; ******
13406                                   <1>
13407 000040C8 30C0                     <1>         xor    al, al ; 0
13408 000040CA 80F920                   <1>         cmp    cl, 32
13409 000040CD 7202                     <1>         jb     short sml_5
13410 000040CF B002                     <1>         mov    al, VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
13411                                   <1> sml_5:
13412 000040D1 AA                       <1>         stosb  ; DirectColorModeInfo
13413                                   <1>
13414                                   <1>         ; // Mandatory information for VBE 2.0 and above
13415                                   <1>
13416 000040D2 B8000000E0               <1>         mov    eax, VBE_DISPI_LFB_PHYSICAL_ADDRESS
13417 000040D7 AB                       <1>         stosd  ; PhysBasePtr
13418 000040D8 29C0                     <1>         sub    eax, eax
13419 000040DA AB                       <1>         stosd  ; OffScreenMemOffset = 0
13420 000040DB 66AB                     <1>         stosw  ; OffScreenMemSize = 0
13421                                   <1>
13422                                   <1>         ;// Mandatory information for VBE 3.0 and above
13423                                   <1>
13424                                   <1>         ; ebx = pitch
13425 000040DD 6689D8                  <1>         mov    ax, bx
13426                                   <1>         ;stosw
13427                                   <1>
13428                                   <1>         ;xor   al, al
13429                                   <1>         ;stosb ; BnkNumberOfPages = 0
13430                                   <1>         ;stosb ; LinNumberOfPages = 0
13431                                   <1>
13432 000040E0 AB                       <1>         stosd  ; pitch (word), 0 (byte), 0 (byte)
13433                                   <1>
13434 000040E1 AD                       <1>         lodsd  ; LinRedMaskSize (AL), LinRedFieldPosition (AH)
13435                                   <1>                ; LinGreenMaskSize (16), LinGreenFieldPosition (24)
13436 000040E2 AB                       <1>         stosd
13437 000040E3 AD                       <1>         lodsd  ; LinBlueMaskSize (AL), LinBlueFieldPosition (AH)
13438                                   <1>                ; LinRsvdMaskSize (16), LinRsvdFieldPosition (24)
13439 000040E4 AB                       <1>         stosd
13440                                   <1>
13441 000040E5 29C0                     <1>         sub    eax, eax
13442 000040E7 AB                       <1>         stosd  ; MaxPixelClock = 0
13443                                   <1>
13444                                   <1>         ;mov   eax, MODE_INFO_LIST
13445                                   <1>         ; 11/12/2020
13446 000040E8 BE[22120300]             <1>         mov    esi, MODE_INFO_LIST
13447                                   <1>
13448 000040ED C3                       <1>         retn
13449                                   <1>
13450                                   <1> ; end of set_mode_info_list ; edi = set_mode_info_list + 68
13451                                   <1>
13452                                   <1> pci_get_lfb_addr:
13453                                   <1>         ; 11/12/2020
13454                                   <1>         ; Get linear frame buffer base from PCI
13455                                   <1>         ; (vgabios.c, 02/01/2020, vruppert)
13456                                   <1>         ;
13457                                   <1>         ; Input:
13458                                   <1>         ;      ax = PCI device vendor id
13459                                   <1>         ; Output:
13460                                   <1>         ;      ax  = LFB address (high 16 bit) (zf=0)
13461                                   <1>         ;      eax = 0 -> not found (error) (zf=1)
13462                                   <1>         ;
13463                                   <1>         ; Modified registers: eax
13464                                   <1>
13465 000040EE 53                       <1>         push   ebx
13466 000040EF 51                       <1>         push   ecx
13467 000040F0 52                       <1>         push   edx
13468                                   <1>         ;
13469 000040F1 89C3                     <1>         mov    ebx, eax
13470 000040F3 31C9                     <1>         xor    ecx, ecx
13471 000040F5 28D2                     <1>         sub    dl, dl ; mov dl, 0
13472 000040F7 E842000000               <1>         call   pci_read_reg
13473 000040FC 6683F8FF                 <1>         cmp    ax, 0FFFFh
13474 00004100 7417                     <1>         je     short pci_get_lfb_addr_fail
13475                                   <1> pci_get_lfb_addr_next_dev:
13476 00004102 28D2                     <1>         sub    dl, dl ; mov dl, 0
13477 00004104 E835000000               <1>         call   pci_read_reg
13478 00004109 6639D8                   <1>         cmp    ax, bx ; check vendor
13479 0000410C 740F                     <1>         je     short pci_get_lfb_addr_found
13480 0000410E 6683C108                 <1>         add    cx, 08h
13481 00004112 6681F90002               <1>         cmp    cx, 200h ; search bus 0 and 1
13482 00004117 72E9                     <1>         jb     short pci_get_lfb_addr_next_dev
13483                                   <1> pci_get_lfb_addr_fail:
13484 00004119 31C0                     <1>         xor    eax, eax ; no LFB
13485                                   <1>         ; zf = 1
13486 0000411B EB1D                     <1>         jmp    short pci_get_lfb_addr_return
13487                                   <1> pci_get_lfb_addr_found:
13488 0000411D B210                     <1>         mov    dl, 10h      ; I/O space 0
13489 0000411F E81A000000               <1>         call   pci_read_reg
13490 00004124 66A9F1FF                 <1>         test   ax, 0FFF1h
13491 00004128 740D                     <1>         jz     short pci_get_lfb_addr_success
13492 0000412A B214                     <1>         mov    dl, 14h ; I/O space 1
```

```
13493 0000412C E80D000000          <1>       call   pci_read_reg
13494 00004131 66A9F1FF            <1>       test   ax, 0FFF1h
13495 00004135 75E2                <1>       jnz    short pci_get_lfb_addr_fail
13496                              <1> pci_get_lfb_addr_success:
13497 00004137 C1E810              <1>       shr    eax, 16 ; LFB address (hw)
13498                              <1>       ; zf = 0
13499                              <1> pci_get_lfb_addr_return:
13500 0000413A 5A                  <1>       pop    edx
13501 0000413B 59                  <1>       pop    ecx
13502 0000413C 5B                  <1>       pop    ebx
13503 0000413D C3                  <1>       retn
13504                              <1>
13505                              <1> pci_read_reg:
13506                              <1>       ; 11/12/2020
13507                              <1>       ; Read PCI register
13508                              <1>       ; (vgabios.c, 02/01/2020, vruppert)
13509                              <1>       ;
13510                              <1>       ; Input:
13511                              <1>       ;      cx = device/function
13512                              <1>       ;      dl = register
13513                              <1>       ; Output:
13514                              <1>       ;      eax = value
13515                              <1>       ;
13516                              <1>       ; Modified registers: eax, edx
13517                              <1>
13518 0000413E B800008000          <1>       mov    eax, 00800000h
13519 00004143 6689C8              <1>       mov    ax, cx
13520 00004146 C1E008              <1>       shl    eax, 8
13521 00004149 88D0                <1>       mov    al, dl
13522 0000414B 66BAF80C            <1>       mov    dx, 0CF8h
13523 0000414F EF                  <1>       out    dx, eax
13524 00004150 80C204              <1>       add    dl, 4 ; mov dx, 0CFCh
13525 00004153 ED                  <1>       in     eax, dx
13526 00004154 C3                  <1>       retn
13527                              <1>
13528                              <1> %endif
13529                              <1>
13530                              <1> ; -----------
13531                              <1>
13532                              <1> %if 0
13533                              <1>
13534                              <1> mode_info_find_mode:
13535                              <1>       ; 25/11/2020
13536                              <1>       ; Input:
13537                              <1>       ;      bx = VESA VBE2 video mode (+ bochs extensions)
13538                              <1>       ; Output:
13539                              <1>       ;      esi = mode info address (for BX input)
13540                              <1>       ;      esi = 0 -> not found
13541                              <1>       ;
13542                              <1>       ; Modified registers: eax, esi
13543                              <1>
13544                              <1>       xor    eax, eax
13545                              <1>       mov    esi, MODE_INFO_LIST
13546                              <1> mifm_1:
13547                              <1>       mov    ax, [esi]
13548                              <1>       cmp    ax, bx
13549                              <1>       je     short mifm_2
13550                              <1>       add    esi, MODEINFO.size ; add esi, 68
13551                              <1>       cmp    esi, VBE_VESA_MODE_END_OF_LIST
13552                              <1>       jb     short mifm_1
13553                              <1>       ; not found
13554                              <1>       sub    esi, esi ; 0
13555                              <1> mifm_2
13556                              <1>       retn
13557                              <1>
13558                              <1> dispi_get_bpp:
13559                              <1>       ; 28/11/2020
13560                              <1>       ; Input:
13561                              <1>       ;      none
13562                              <1>       ; Output:
13563                              <1>       ;      al = Bits per pixel
13564                              <1>       ;           (8,16,24,32)
13565                              <1>       ;      ah = Bytes per pixel
13566                              <1>       ;           (1,2,3,4)
13567                              <1>       ;
13568                              <1>       ; Modified registers: none
13569                              <1>
13570                              <1>       ;push  edx
13571                              <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
13572                              <1>       ;mov   ax, 03h        ; VBE_DISPI_INDEX_BPP
13573                              <1>       ;out   dx, ax
13574                              <1>       ;;mov  dx, 01CFh ; VBE_DISPI_IOPORT_DATA
13575                              <1>       ;;mov  dl, 0CFh
13576                              <1>       ;inc   dl
13577                              <1>       ;in    ax, dx
13578                              <1>       ;mov   ah, al
13579                              <1>       ;shr   ah, 3 ; / 8
13580                              <1>       ;;test al, 7 ; 15 bit graphics mode
13581                              <1>       ;;jz   short get_bpp_noinc
13582                              <1>       ;;inc  ah
13583                              <1> ;;get_bpp_noinc:
13584                              <1>       ;pop   edx
13585                              <1>       ;retn
13586                              <1>
13587                              <1>       ; 28/11/2020
13588                              <1>       ; Modified registers: edx
13589                              <1>       ;push  edx
13590                              <1>       mov    dx, 03h ; VBE_DISPI_INDEX_BPP
13591                              <1>       call   dispi_get_parms
13592                              <1>       ;pop   edx
13593                              <1>       ;retn
13594                              <1>       mov    ah, al
13595                              <1>       shr    ah, 3 ; / 8
13596                              <1>       ;test  al, 7 ; 15 bit graphips mode
13597                              <1>       ;jz    short get_bpp_noinc
```

```
13598                           <1>       ;inc   ah
13599                           <1> ;get_bpp_noinc:
13600                           <1>       ;pop   edx
13601                           <1>       retn
13602                           <1>
13603                           <1> restore_vesa_video_state:
13604                           <1>       ; 14/01/2021
13605                           <1>       ; 06/12/2020
13606                           <1>       ; Input:
13607                           <1>       ;      [vbe3stbufsize] <= 32 ; <= 32*64 bytes
13608                           <1>       ; Output:
13609                           <1>       ;      AX = 004Fh (successed)
13610                           <1>       ;
13611                           <1>       ;      eax = 0 -> buffer size problem
13612                           <1>       ;      eax > 0 and ax <> 004Fh -> failed
13613                           <1>       ;
13614                           <1>       ; Modified regs: eax, ebx, ecx, edx, esi, edi
13615                           <1>
13616                           <1>       ;movzx ecx, word [vbe3stbufsize]
13617                           <1>       ;cmp   cx, 32 ; 32 * 64 bytes
13618                           <1>       ;ja    short r_v_b_s_fail
13619                           <1>
13620                           <1>       movzx  ecx, byte [vbe3stbufsize]; <=32
13621                           <1>       shl    cx, 4 ; dword count for movsd
13622                           <1>       mov    edi, VBE3SAVERESTOREBLOCK ; destination
13623                           <1>                                ; (vbe3 pmi buff)
13624                           <1>       mov    esi, VBE3VIDEOSTATE ; source (kernel buff)
13625                           <1>       rep    movsd
13626                           <1>
13627                           <1>       mov    ax, 4F04h
13628                           <1>       mov    dl, 02h ; restore
13629                           <1>       ;mov   cx, 0Fh
13630                           <1>       mov    cl, 0Fh
13631                           <1>       xor    ebx, ebx ; points to VBE3SAVERESTOREBLOCK
13632                           <1>       jmp    short int10h_32bit_pmi
13633                           <1>
13634                           <1> ;s_v_b_s_fail:
13635                           <1> ;r_v_b_s_fail:
13636                           <1> ;     xor    eax, eax
13637                           <1> ;     retn
13638                           <1>
13639                           <1> save_vesa_video_state:
13640                           <1>       ; 14/01/2021
13641                           <1>       ; 06/12/2020
13642                           <1>       ; Input:
13643                           <1>       ;      [vbe3stbufsize] <= 32 ; <= 32*64 bytes
13644                           <1>       ; Output:
13645                           <1>       ;      AX = 004Fh (successed)
13646                           <1>       ;
13647                           <1>       ;      eax = 0 -> buffer size problem
13648                           <1>       ;      eax > 0 and ax <> 004Fh -> failed
13649                           <1>       ;
13650                           <1>       ; Modified regs: eax, ebx, ecx, edx, esi, edi
13651                           <1>
13652                           <1>       ;cmp   word [vbe33stbufsize], 32
13653                           <1>       ;                  ; 32 * 64 bytes
13654                           <1>       ;ja    short s_v_b_s_fail
13655                           <1>
13656                           <1>       mov    ax, 4F04h
13657                           <1>       mov    dl, 01h ; save
13658                           <1>       ;mov   cx, 0Fh
13659                           <1>       mov    cl, 0Fh
13660                           <1>       xor    ebx, ebx ; points to VBE3SAVERESTOREBLOCK
13661                           <1>
13662                           <1>       call   int10h_32bit_pmi
13663                           <1>
13664                           <1>       movzx  ecx, byte [vbe3stbufsize]; <=32
13665                           <1>       shl    cx, 4 ; dword count for movsd
13666                           <1>       mov    esi, VBE3SAVERESTOREBLOCK ; destination
13667                           <1>                                ; (vbe3 pmi buff)
13668                           <1>       mov    edi, VBE3VIDEOSTATE ; source (kernel buff)
13669                           <1>       rep    movsd
13670                           <1>       retn
13671                           <1>
13672                           <1> dispi_set_bank_farcall:
13673                           <1>       ; 12/04/2021 (32 bit push/pop)
13674                           <1>       ; 11/12/2020
13675                           <1>       ; (This may be 'sysvideo' function, later)
13676                           <1>       ;
13677                           <1>       ; Input:
13678                           <1>       ;      bx = 0000h, set bank number
13679                           <1>       ;         = 0100h, get bank number
13680                           <1>       ;      dx = bank number (if bx = 0)
13681                           <1>       ; Output:
13682                           <1>       ;      dx = bank number
13683                           <1>
13684                           <1>       cmp    bx, 0100h
13685                           <1>       je     short dispi_set_bank_farcall_get
13686                           <1>       or     bx, bx
13687                           <1>       jnz    dispi_set_bank_farcall_error
13688                           <1>       mov    ax, dx
13689                           <1>       ;push  dx
13690                           <1>       ;push  ax
13691                           <1>       ; 12/04/2021
13692                           <1>       push   edx
13693                           <1>       push   eax
13694                           <1>       mov    ax, VBE_DISPI_INDEX_BANK
13695                           <1>       mov    dx, VBE_DISPI_IOPORT_INDEX
13696                           <1>       out    dx, ax
13697                           <1>       ;pop   ax
13698                           <1>       ; 12/04/2021
13699                           <1>       pop    eax
13700                           <1>       mov    dx, VBE_DISPI_IOPORT_DATA
13701                           <1>       out    dx, ax
13702                           <1>       in     ax, dx
```

```
13703                           <1>       ;pop   dx
13704                           <1>       ; 12/04/2021
13705                           <1>       pop    edx
13706                           <1>       cmp    dx, ax
13707                           <1>       jne    short dispi_set_bank_farcall_error
13708                           <1>       mov    ax, 004Fh
13709                           <1>       retn   ; retf for real mode far call
13710                           <1> dispi_set_bank_farcall_get:
13711                           <1>       mov    ax, VBE_DISPI_INDEX_BANK
13712                           <1>       mov    dx, VBE_DISPI_IOPORT_INDEX
13713                           <1>       out    dx, ax
13714                           <1>       mov    dx, VBE_DISPI_IOPORT_DATA
13715                           <1>       in     ax, dx
13716                           <1>       mov    dx, ax
13717                           <1>       retn   ; retf for real mode far call
13718                           <1> dispi_set_bank_farcall_error:
13719                           <1>       mov    ax, 014Fh
13720                           <1>       retn   ; retf for real mode far call
13721                           <1>
13722                           <1> %endif
13723                           <1>
13724                           <1> ; % include 'vidata.s' ; VIDEO DATA
13725                           <1>
13726                           <1> ; /// End Of VIDEO FUNCTIONS ///
 2680
 2681                               setup_rtc_int:
 2682                               ; source: http://wiki.osdev.org/RTC
 2683 00004155 FA                        cli          ; disable interrupts
 2684                                    ; default int frequency is 1024 Hz (Lower 4 bits of register A is 0110b or 6)
 2685                                    ; in order to change this ...
 2686                                    ; frequency  = 32768 >> (rate-1) --> 32768 >> 5 = 1024
 2687                                    ; (rate must be above 2 and not over 15)
 2688                                    ; new rate = 15 --> 32768 >> (15-1) = 2 Hz
 2689 00004156 B08A                     mov    al, 8Ah
 2690 00004158 E670                     out    70h, al ; set index to register A, disable NMI
 2691 0000415A 90                       nop
 2692 0000415B E471                     in     al, 71h ; get initial value of register A
 2693 0000415D 88C4                     mov    ah, al
 2694 0000415F 80E4F0                   and    ah, 0F0h
 2695 00004162 B08A                     mov    al, 8Ah
 2696 00004164 E670                     out    70h, al ; reset index to register A
 2697 00004166 88E0                     mov    al, ah
 2698 00004168 0C0F                     or     al, 0Fh      ; new rate (0Fh -> 15)
 2699 0000416A E671                     out    71h, al ; write only our rate to A. Note, rate is the bottom 4 bits.
 2700                                   ; enable RTC interrupt
 2701 0000416C B08B                     mov    al, 8Bh ;
 2702 0000416E E670                     out    70h, al ; select register B and disable NMI
 2703 00004170 90                       nop
 2704 00004171 E471                     in     al, 71h ; read the current value of register B
 2705 00004173 88C4                     mov    ah, al  ;
 2706 00004175 B08B                     mov    al, 8Bh ;
 2707 00004177 E670                     out    70h, al ; set the index again (a read will reset the index to register B)
 2708 00004179 88E0                     mov    al, ah  ;
 2709 0000417B 0C40                     or     al, 40h ;
 2710 0000417D E671                     out    71h, al ; write the previous value ORed with 0x40. This turns on bit 6 of register B
 2711 0000417F FB                       sti
 2712 00004180 C3                       retn
 2713
 2714                               ; Write memory information
 2715                               ; 29/01/2016
 2716                               ; 06/11/2014
 2717                               ; 14/08/2015
 2718                               memory_info:
 2719 00004181 A1[AC800100]             mov    eax, [memory_size] ; in pages
 2720 00004186 50                       push   eax
 2721 00004187 C1E00C                   shl    eax, 12                ; in bytes
 2722 0000418A BB0A000000               mov    ebx, 10
 2723 0000418F 89D9                     mov    ecx, ebx        ; 10
 2724 00004191 BE[31430100]             mov    esi, mem_total_b_str
 2725 00004196 E8D3000000               call   bintdstr
 2726 0000419B 58                       pop    eax
 2727 0000419C B107                     mov    cl, 7
 2728 0000419E BE[55430100]             mov    esi, mem_total_p_str
 2729 000041A3 E8C6000000               call   bintdstr
 2730                                   ; 14/08/2015
 2731 000041A8 E8DE000000               call   calc_free_mem
 2732                                   ; edx = calculated free pages
 2733                                   ; ecx = 0
 2734 000041AD A1[B0800100]             mov    eax, [free_pages]
 2735 000041B2 39D0                     cmp    eax, edx ; calculated free mem value
 2736                                          ; and initial free mem value are same or not?
 2737 000041B4 751D                     jne    short pmim ; print mem info with '?' if not
 2738 000041B6 52                       push   edx ; free memory in pages
 2739                                   ;mov   eax, edx
 2740 000041B7 C1E00C                   shl    eax, 12 ; convert page count
 2741                                                  ; to byte count
 2742 000041BA B10A                     mov    cl, 10
 2743 000041BC BE[75430100]             mov    esi, free_mem_b_str
 2744 000041C1 E8A8000000               call   bintdstr
 2745 000041C6 58                       pop    eax
 2746 000041C7 B107                     mov    cl, 7
 2747 000041C9 BE[99430100]             mov    esi, free_mem_p_str
 2748 000041CE E89B000000               call   bintdstr
 2749                               pmim:
 2750 000041D3 BE[1F430100]             mov    esi, msg_memory_info
 2751                                   ;
 2752 000041D8 B407                     mov    ah, 07h ; Black background,
 2753                                                  ; light gray forecolor
 2754                               print_kmsg: ; 29/01/2016
 2755 000041DA 8825[D7800100]           mov    [ccolor], ah
 2756                               pkmsg_loop:
 2757 000041E0 AC                       lodsb
 2758 000041E1 08C0                     or     al, al
 2759 000041E3 7410                     jz     short pkmsg_ok
 2760 000041E5 56                       push   esi
```

```
                                        ; 13/05/2016
000041E6 0FB61D[D7800100]               movzx  ebx, byte [ccolor]
                                                ; Video page 0 (bh=0)
000041ED E871E0FFFF                     call   _write_tty
000041F2 5E                             pop    esi
000041F3 EBEB                           jmp    short pkmsg_loop
                                pkmsg_ok:
000041F5 C3                             retn

                                ; 19/12/2020
                                ; temporary
                                ; Write default liner frame buffer address
                                ;
                                default_lfb_info:
000041F6 66A1[D90E0000]                 mov    ax, [def_LFB_addr] ; high word
000041FC E829000000                     call   wordtohex
00004201 A3[F1430100]                   mov    dword [lfb_addr_str], eax
00004206 BE[DA430100]                   mov    esi, msg_lfb_addr
0000420B B40F                           mov    ah, 0Fh ; Black background,
                                                ; white forecolor
0000420D EBCB                           jmp    short print_kmsg

                                ; Convert binary number to hexadecimal string
                                ; 10/05/2015
                                ; dsectpm.s (28/02/2015)
                                ; Retro UNIX 386 v1 - Kernel v0.2.0.6
                                ; 01/12/2014
                                ; 25/11/2014
                                ;
                                bytetohex:
                                        ; INPUT ->
                                        ;      AL = byte (binary number)
                                        ; OUTPUT ->
                                        ;      AX = hexadecimal string
                                        ;
0000420F 53                             push   ebx
00004210 31DB                           xor    ebx, ebx
00004212 88C3                           mov    bl, al
00004214 C0EB04                         shr    bl, 4
00004217 8A9B[5D420000]                 mov    bl, [ebx+hexchrs]
0000421D 86D8                           xchg   bl, al
0000421F 80E30F                         and    bl, 0Fh
00004222 8AA3[5D420000]                 mov    ah, [ebx+hexchrs]
00004228 5B                             pop    ebx
00004229 C3                             retn

                                wordtohex:
                                        ; INPUT ->
                                        ;      AX = word (binary number)
                                        ; OUTPUT ->
                                        ;      EAX = hexadecimal string
                                        ;
0000422A 53                             push   ebx
0000422B 31DB                           xor    ebx, ebx
0000422D 86E0                           xchg   ah, al
0000422F 6650                           push   ax ; * save ax
00004231 88E3                           mov    bl, ah
00004233 C0EB04                         shr    bl, 4
00004236 8A83[5D420000]                 mov    al, [ebx+hexchrs]
0000423C 88E3                           mov    bl, ah
0000423E 80E30F                         and    bl, 0Fh
00004241 8AA3[5D420000]                 mov    ah, [ebx+hexchrs]
00004247 C1E010                         shl    eax, 16 ; ax -> hw of eax
0000424A 6658                           pop    ax ; * restore ax
0000424C 5B                             pop    ebx
0000424D EBC0                           jmp    short bytetohex
                                        ;mov   bl, al
                                        ;shr   bl, 4
                                        ;mov   bl, [ebx+hexchrs]
                                        ;xchg  bl, al
                                        ;and   bl, 0Fh
                                        ;mov   ah, [ebx+hexchrs]
                                        ;pop   ebx
                                        ;retn

                                dwordtohex:
                                        ; INPUT ->
                                        ;      EAX = dword (binary number)
                                        ; OUTPUT ->
                                        ;      EDX:EAX = hexadecimal string
                                        ;
0000424F 50                             push   eax
00004250 C1E810                         shr    eax, 16
00004253 E8D2FFFFFF                     call   wordtohex
00004258 89C2                           mov    edx, eax
0000425A 58                             pop    eax
                                        ;call  wordtohex
                                        ;retn
                                        ; 18/04/2021
0000425B EBCD                           jmp    short wordtohex

                                ; 10/05/2015
                                hex_digits:
                                hexchrs:
0000425D 303132333435363738-            db '0123456789ABCDEF'
00004266 39414243444546
                                ; 19/01/2021 - VESA EDID ready flag (4Fh)
0000426D 00                     edid: db 0

                                ; Convert binary number to decimal/numeric string
                                ; 06/11/2014
                                ; Temporary Code
                                ;

                                bintdstr:
```

```
                                        ; EAX = binary number
                                        ; ESI = decimal/numeric string address
                                        ; EBX = divisor (10)
                                        ; ECX = string length (<=10)
0000426E 01CE                           add    esi, ecx
                                    btdstr0:
00004270 4E                             dec    esi
00004271 31D2                           xor    edx, edx
00004273 F7F3                           div    ebx
00004275 80C230                         add    dl, 30h
00004278 8816                           mov    [esi], dl
0000427A FEC9                           dec    cl
0000427C 740C                           jz     short btdstr2 ; 08/09/2016
0000427E 09C0                           or     eax, eax
00004280 75EE                           jnz    short btdstr0
                                    btdstr1:
00004282 4E                             dec    esi
00004283 C60620                         mov   byte [esi], 20h ; blank space
00004286 FEC9                           dec    cl
00004288 75F8                           jnz    short btdstr1
                                    btdstr2:
0000428A C3                             retn

                                    ; Calculate free memory pages on M.A.T.
                                    ; 06/11/2014
                                    ; Temporary Code
                                    ;

                                    calc_free_mem:
0000428B 31D2                           xor    edx, edx
                                        ;xor   ecx, ecx
0000428D 668B0D[C0800100]               mov    cx, [mat_size] ; in pages
00004294 C1E10A                         shl    ecx, 10      ; 1024 dwords per page
00004297 BE00001000                     mov    esi, MEM_ALLOC_TBL
                                    cfm0:
0000429C AD                             lodsd
0000429D 51                             push   ecx
0000429E B920000000                     mov    ecx, 32
                                    cfm1:
000042A3 D1E8                           shr    eax, 1
000042A5 7301                           jnc    short cfm2
000042A7 42                             inc    edx
                                    cfm2:
000042A8 E2F9                           loop   cfm1
000042AA 59                             pop    ecx
000042AB E2EF                           loop   cfm0
000042AD C3                             retn

                                    %include 'diskio.s'  ; 07/03/2015
                                <1> ; ****************************************************************************
                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.4 - diskio.s
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Last Update: 18/04/2021
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Beginning: 24/01/2016
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Assembler: NASM version 2.15 (trdos386.s)
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Turkish Rational DOS
                                <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
                                <1> ;
                                <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
                                <1> ; diskio.inc (22/08/2015)
                                <1> ;
                                <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
                                <1> ; ****************************************************************************
                                <1>
                                <1> ; Retro UNIX 386 v1 Kernel - DISKIO.INC
                                <1> ; Last Modification: 22/08/2015
                                <1> ;     (Initialized Disk Parameters Data is in 'DISKDATA.INC')
                                <1> ;     (Uninitialized Disk Parameters Data is in 'DISKBSS.INC')
                                <1>
                                <1> ; DISK I/O SYSTEM - Erdogan Tan (Retro UNIX 386 v1 project)
                                <1>
                                <1> ; ///////// DISK I/O SYSTEM ///////////////
                                <1>
                                <1> ; 11/04/2021
                                <1> ;; 06/02/2015
                                <1> ;diskette_io:
                                <1>       ;clc ; 20/07/2020
                                <1>       ;pushfd
                                <1>       ;push  cs
                                <1>       ;;call       DISKETTE_IO_1
                                <1>       ;;retn
                                <1>
                                <1> ;;;;;; DISKETTE I/O ;;;;;;;;;;;;;;;;;;;; 06/02/2015 ;;;
                                <1> ;////////////////////////////////////////////////////
                                <1>
                                <1> ; DISKETTE I/O - Erdogan Tan (Retro UNIX 386 v1 project)
                                <1> ; 20/02/2015
                                <1> ; 06/02/2015 (unix386.s)
                                <1> ; 16/12/2014 - 02/01/2015 (dsectrm2.s)
                                <1> ;
                                <1> ; Code (DELAY) modifications - AWARD BIOS 1999 (ADISK.EQU, COMMON.MAC)
                                <1> ;
                                <1> ; ADISK.EQU
                                <1>
                                <1> ;----- Wait control constants
                                <1>
                                <1> ;amount of time to wait while RESET is active.
                                <1>
                                <1> WAITCPU_RESET_ON   EQU    21           ;Reset on must last at least 14us
                                <1>                                        ;at 250 KBS xfer rate.
                                <1>                                        ;see INTEL MCS, 1985, pg. 5-456
                                <1>
```

```
2970                            <1> WAITCPU_FOR_STATUS EQU    100           ;allow 30 microseconds for
2971                            <1>                                         ;status register to become valid
2972                            <1>                                         ;before re-reading.
2973                            <1>
2974                            <1> ;After sending a byte to NEC, status register may remain
2975                            <1> ;incorrectly set for 24 us.
2976                            <1>
2977                            <1> WAITCPU_RQM_LOW    EQU    24           ;number of loops to check for
2978                            <1>                                         ;RQM low.
2979                            <1>
2980                            <1> ; COMMON.MAC
2981                            <1> ;
2982                            <1> ;     Timing macros
2983                            <1> ;
2984                            <1>
2985                            <1> %macro      SIODELAY 0                  ; SHORT IODELAY
2986                            <1>             jmp short $+2
2987                            <1> %endmacro
2988                            <1>
2989                            <1> %macro      IODELAY  0                  ; NORMAL IODELAY
2990                            <1>             jmp short $+2
2991                            <1>             jmp short $+2
2992                            <1> %endmacro
2993                            <1>
2994                            <1> %macro      NEWIODELAY 0
2995                            <1>             out    0ebh,al
2996                            <1> %endmacro
2997                            <1>
2998                            <1> ; (According to) AWARD BIOS 1999 - ATORGS.ASM (dw -> equ, db -> equ)
2999                            <1> ;;; WAIT_FOR_MEM
3000                            <1> ;WAIT_FDU_INT_LO   equ    017798        ; 2.5 secs in 30 micro units.
3001                            <1> ;WAIT_FDU_INT_HI   equ    1
3002                            <1> WAIT_FDU_INT_LH    equ    83334         ; 27/02/2015 (2.5 seconds waiting)
3003                            <1> ;;; WAIT_FOR_PORT
3004                            <1> ;WAIT_FDU_SEND_LO  equ    16667         ; .5 secons in 30 us units.
3005                            <1> ;WAIT_FDU_SEND_HI  equ    0
3006                            <1> WAIT_FDU_SEND_LH   equ    16667         ; 27/02/2015
3007                            <1> ;Time to wait while waiting for each byte of NEC results = .5
3008                            <1> ;seconds.  .5 seconds = 500,000 micros.  500,000/30 = 16,667.
3009                            <1> ;WAIT_FDU_RESULTS_LO      equ    16667  ; .5 seconds in 30 micro units.
3010                            <1> ;WAIT_FDU_RESULTS_HI      equ    0
3011                            <1> WAIT_FDU_RESULTS_LH       equ    16667  ; 27/02/2015
3012                            <1> ;;; WAIT_REFRESH
3013                            <1> ;amount of time to wait for head settle, per unit in parameter
3014                            <1> ;table = 1 ms.
3015                            <1> WAIT_FDU_HEAD_SETTLE      equ    33     ; 1 ms in 30 micro units.
3016                            <1>
3017                            <1>
3018                            <1> ; ///////////////// DISKETTE I/O ////////////////
3019                            <1>
3020                            <1> ; 11/12/2014 (copy from IBM PC-XT Model 286 BIOS - POSTEQU.INC)
3021                            <1>
3022                            <1> ;--------------------------------------
3023                            <1> ;     EQUATES USED BY POST AND BIOS    :
3024                            <1> ;--------------------------------------
3025                            <1>
3026                            <1> ;--------- 8042 KEYBOARD INTERFACE AND DIAGNOSTIC CONTROL REGISTERS ------------
3027                            <1> ;PORT_A            EQU    060H   ; 8042 KEYBOARD SCAN CODE/CONTROL PORT
3028                            <1> ;PORT_B            EQU    061H   ; PORT B READ/WRITE DIAGNOSTIC REGISTER
3029                            <1> ;REFRESH_BIT EQU    00010000B    ; REFRESH TEST BIT
3030                            <1>
3031                            <1> ;--------------------------------------
3032                            <1> ;     CMOS EQUATES FOR THIS SYSTEM     :
3033                            <1> ;-------------------------------------------------------------------------
3034                            <1> ;CMOS_PORT   EQU    070H         ; I/O ADDRESS OF CMOS ADDRESS PORT
3035                            <1> ;CMOS_DATA   EQU    071H         ; I/O ADDRESS OF CMOS DATA PORT
3036                            <1> ;NMI         EQU    10000000B    ; DISABLE NMI INTERRUPTS MASK -
3037                            <1>                                 ;  HIGH BIT OF CMOS LOCATION ADDRESS
3038                            <1>
3039                            <1> ;---------- CMOS TABLE LOCATION ADDRESS'S ## ---------------------------------
3040                            <1> CMOS_DISKETTE      EQU    010H   ; DISKETTE DRIVE TYPE BYTE       ;
3041                            <1> ;           EQU    011H         ; - RESERVED                     ;C
3042                            <1> CMOS_DISK    EQU    012H         ; FIXED DISK TYPE BYTE           ;H
3043                            <1> ;           EQU    013H         ; - RESERVED                     ;E
3044                            <1> CMOS_EQUIP   EQU    014H         ; EQUIPMENT WORD LOW BYTE        ;C
3045                            <1>
3046                            <1> ;---------- DISKETTE EQUATES --------------------------------------------------
3047                            <1> INT_FLAG     EQU    10000000B    ; INTERRUPT OCCURRENCE FLAG
3048                            <1> DSK_CHG      EQU    10000000B    ; DISKETTE CHANGE FLAG MASK BIT
3049                            <1> DETERMINED   EQU    00010000B    ; SET STATE DETERMINED IN STATE BITS
3050                            <1> HOME         EQU    00010000B    ; TRACK 0 MASK
3051                            <1> SENSE_DRV_ST EQU    00000100B    ; SENSE DRIVE STATUS COMMAND
3052                            <1> TRK_SLAP     EQU    030H         ; CRASH STOP (48 TPI DRIVES)
3053                            <1> QUIET_SEEK   EQU    00AH         ; SEEK TO TRACK 10
3054                            <1> ;MAX_DRV     EQU    2            ; MAX NUMBER OF DRIVES
3055                            <1> HD12_SETTLE  EQU    15           ; 1.2 M HEAD SETTLE TIME
3056                            <1> HD320_SETTLE EQU    20           ; 320 K HEAD SETTLE TIME
3057                            <1> MOTOR_WAIT   EQU    37           ; 2 SECONDS OF COUNTS FOR MOTOR TURN OFF
3058                            <1>
3059                            <1> ;---------- DISKETTE ERRORS ---------------------------------------------------
3060                            <1> ;TIME_OUT    EQU    080H         ; ATTACHMENT FAILED TO RESPOND
3061                            <1> ;BAD_SEEK    EQU    040H         ; SEEK OPERATION FAILED
3062                            <1> BAD_NEC      EQU    020H         ; DISKETTE CONTROLLER HAS FAILED
3063                            <1> BAD_CRC      EQU    010H         ; BAD CRC ON DISKETTE READ
3064                            <1> MED_NOT_FND  EQU    00CH         ; MEDIA TYPE NOT FOUND
3065                            <1> DMA_BOUNDARY EQU    009H         ; ATTEMPT TO DMA ACROSS 64K BOUNDARY
3066                            <1> BAD_DMA      EQU    008H         ; DMA OVERRUN ON OPERATION
3067                            <1> MEDIA_CHANGE EQU    006H         ; MEDIA REMOVED ON DUAL ATTACH CARD
3068                            <1> RECORD_NOT_FND     EQU    004H   ; REQUESTED SECTOR NOT FOUND
3069                            <1> WRITE_PROTECT      EQU    003H   ; WRITE ATTEMPTED ON WRITE PROTECT DISK
3070                            <1> BAD_ADDR_MARK      EQU    002H   ; ADDRESS MARK NOT FOUND
3071                            <1> BAD_CMD      EQU    001H         ; BAD COMMAND PASSED TO DISKETTE I/O
3072                            <1>
3073                            <1> ;---------- DISK CHANGE LINE EQUATES ------------------------------------------
3074                            <1> NOCHGLN      EQU    001H         ; NO DISK CHANGE LINE AVAILABLE
```

```
3075                    <1> CHGLN       EQU    002H         ; DISK CHANGE LINE AVAILABLE
3076                    <1>
3077                    <1> ;---------- MEDIA/DRIVE STATE INDICATORS ----------------------------------------
3078                    <1> TRK_CAPA    EQU    00000001B    ; 80 TRACK CAPABILITY
3079                    <1> FMT_CAPA    EQU    00000010B    ; MULTIPLE FORMAT CAPABILITY (1.2M)
3080                    <1> DRV_DET     EQU    00000100B    ; DRIVE DETERMINED
3081                    <1> MED_DET     EQU    00010000B    ; MEDIA DETERMINED BIT
3082                    <1> DBL_STEP    EQU    00100000B    ; DOUBLE STEP BIT
3083                    <1> RATE_MSK    EQU    11000000B    ; MASK FOR CLEARING ALL BUT RATE
3084                    <1> RATE_500    EQU    00000000B    ; 500 KBS DATA RATE
3085                    <1> RATE_300    EQU    01000000B    ; 300 KBS DATA RATE
3086                    <1> RATE_250    EQU    10000000B    ; 250 KBS DATA RATE
3087                    <1> STRT_MSK    EQU    00001100B    ; OPERATION START RATE MASK
3088                    <1> SEND_MSK    EQU    11000000B    ; MASK FOR SEND RATE BITS
3089                    <1>
3090                    <1> ;---------- MEDIA/DRIVE STATE INDICATORS COMPATIBILITY -------------------------
3091                    <1> M3D3U       EQU    00000000B    ; 360 MEDIA/DRIVE NOT ESTABLISHED
3092                    <1> M3D1U       EQU    00000001B    ; 360 MEDIA,1.2DRIVE NOT ESTABLISHED
3093                    <1> M1D1U       EQU    00000010B    ; 1.2 MEDIA/DRIVE NOT ESTABLISHED
3094                    <1> MED_UNK     EQU    00000111B    ; NONE OF THE ABOVE
3095                    <1>
3096                    <1> ;---------- INTERRUPT EQUATES ---------------------------------------------------
3097                    <1> ;EOI        EQU    020H         ; END OF INTERRUPT COMMAND TO 8259
3098                    <1> ;INTA00            EQU    020H         ; 8259 PORT
3099                    <1> INTA01             EQU    021H         ; 8259 PORT
3100                    <1> INTB00             EQU    0A0H         ; 2ND 8259
3101                    <1> INTB01             EQU    0A1H         ;
3102                    <1>
3103                    <1> ;--------------------------------------------------------------------------------
3104                    <1> DMA08       EQU    008H         ; DMA STATUS REGISTER PORT ADDRESS
3105                    <1> DMA         EQU    000H         ; DMA CH.0 ADDRESS REGISTER PORT ADDRESS
3106                    <1> DMA18       EQU    0D0H         ; 2ND DMA STATUS PORT ADDRESS
3107                    <1> DMA1        EQU    0C0H         ; 2ND DMA CH.0 ADDRESS REGISTER ADDRESS
3108                    <1> ;--------------------------------------------------------------------------------
3109                    <1> ;TIMER             EQU    040H         ; 8254 TIMER - BASE ADDRESS
3110                    <1>
3111                    <1> ;--------------------------------------------------------------------------------
3112                    <1> DMA_PAGE    EQU    081H         ; START OF DMA PAGE REGISTERS
3113                    <1>
3114                    <1> ; 06/02/2015 (unix386.s, protected mode modifications)
3115                    <1> ; (unix386.s <-- dsectrm2.s)
3116                    <1> ; 11/12/2014 (copy from IBM PC-XT Model 286 BIOS - DSEG.INC)
3117                    <1>
3118                    <1> ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
3119                    <1> ; 10/12/2014
3120                    <1> ;
3121                    <1> ;int40h:
3122                    <1> ;   pushf
3123                    <1> ;   push  cs
3124                    <1> ;   ;cli
3125                    <1> ;   call  DISKETTE_IO_1
3126                    <1> ;   retn
3127                    <1>
3128                    <1> ; DSKETTE ----- 04/21/86 DISKETTE BIOS
3129                    <1> ; (IBM PC XT Model 286 System BIOS Source Code, 04-21-86)
3130                    <1> ;
3131                    <1>
3132                    <1> ;-- INT13H ----------------------------------------------------------------------
3133                    <1> ; DISKETTE I/O
3134                    <1> ;   THIS INTERFACE PROVIDES ACCESS TO THE 5 1/4 INCH 360 KB,
3135                    <1> ;   1.2 MB, 720 KB AND 1.44 MB DISKETTE DRIVES.
3136                    <1> ; INPUT
3137                    <1> ;   (AH) =  00H RESET DISKETTE SYSTEM
3138                    <1> ;           HARD RESET TO NEC, PREPARE COMMAND, RECALIBRATE REQUIRED
3139                    <1> ;           ON ALL DRIVES
3140                    <1> ;--------------------------------------------------------------------------------
3141                    <1> ;   (AH)= 01H  READ THE STATUS OF THE SYSTEM INTO (AH)
3142                    <1> ;           @DISKETTE_STATUS FROM LAST OPERATION IS USED
3143                    <1> ;--------------------------------------------------------------------------------
3144                    <1> ;   REGISTERS FOR READ/WRITE/VERIFY/FORMAT
3145                    <1> ;   (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
3146                    <1> ;   (DH) - HEAD NUMBER (0-1 ALLOWED, NOT VALUE CHECKED)
3147                    <1> ;   (CH) - TRACK NUMBER (NOT VALUE CHECKED)
3148                    <1> ;           MEDIA  DRIVE  TRACK NUMBER
3149                    <1> ;           320/360      320/360          0-39
3150                    <1> ;           320/360      1.2M       0-39
3151                    <1> ;           1.2M   1.2M       0-79
3152                    <1> ;           720K   720K       0-79
3153                    <1> ;           1.44M  1.44M      0-79
3154                    <1> ;   (CL) -       SECTOR NUMBER (NOT VALUE CHECKED, NOT USED FOR FORMAT)
3155                    <1> ;           MEDIA  DRIVE  SECTOR NUMBER
3156                    <1> ;           320/360      320/360            1-8/9
3157                    <1> ;           320/360      1.2M        1-8/9
3158                    <1> ;           1.2M   1.2M        1-15
3159                    <1> ;           720K   720K        1-9
3160                    <1> ;           1.44M  1.44M       1-18
3161                    <1> ;   (AL)   NUMBER OF SECTORS (NOT VALUE CHECKED)
3162                    <1> ;           MEDIA  DRIVE  MAX NUMBER OF SECTORS
3163                    <1> ;           320/360      320/360              8/9
3164                    <1> ;           320/360      1.2M           8/9
3165                    <1> ;           1.2M   1.2M         15
3166                    <1> ;           720K   720K         9
3167                    <1> ;           1.44M  1.44M        18
3168                    <1> ;
3169                    <1> ;   (ES:BX) - ADDRESS OF BUFFER (NOT REQUIRED FOR VERIFY)
3170                    <1> ;
3171                    <1> ;--------------------------------------------------------------------------------
3172                    <1> ;   (AH)= 02H  READ THE DESIRED SECTORS INTO MEMORY
3173                    <1> ;--------------------------------------------------------------------------------
3174                    <1> ;   (AH)= 03H  WRITE THE DESIRED SECTORS FROM MEMORY
3175                    <1> ;--------------------------------------------------------------------------------
3176                    <1> ;   (AH)= 04H  VERIFY THE DESIRED SECTORS
3177                    <1> ;--------------------------------------------------------------------------------
3178                    <1> ;   (AH)= 05H  FORMAT THE DESIRED TRACK
3179                    <1> ;           (ES,BX) MUST POINT TO THE COLLECTION OF DESIRED ADDRESS FIELDS
```

```
3180                    <1> ;           FOR THE      TRACK. EACH FIELD IS COMPOSED OF 4 BYTES, (C,H,R,N),
3181                    <1> ;           WHERE C = TRACK NUMBER, H=HEAD NUMBER, R = SECTOR NUMBER,
3182                    <1> ;           N= NUMBER OF BYTES PER SECTOR (00=128,01=256,02=512,03=1024),
3183                    <1> ;           THERE MUST BE ONE ENTRY FOR EVERY SECTOR ON THE TRACK.
3184                    <1> ;           THIS INFORMATION IS USED TO FIND THE REQUESTED SECTOR DURING
3185                    <1> ;           READ/WRITE ACCESS.
3186                    <1> ;           PRIOR TO FORMATTING A DISKETTE, IF THERE EXISTS MORE THAN
3187                    <1> ;           ONE SUPPORTED MEDIA FORMAT TYPE WITHIN THE DRIVE IN QUESTION,
3188                    <1> ;           THEN "SET DASD TYPE" (INT 13H, AH = 17H) OR 'SET MEDIA TYPE'
3189                    <1> ;           (INT 13H, AH =  18H) MUST BE CALLED TO SET THE DISKETTE TYPE
3190                    <1> ;           THAT IS TO BE FORMATTED. IF "SET DASD TYPE" OR "SET MEDIA TYPE"
3191                    <1> ;           IS NOT CALLED, THE FORMAT ROUTINE WILL ASSUME THE
3192                    <1> ;           MEDIA FORMAT TO BE THE MAXIMUM CAPACITY OF THE DRIVE.
3193                    <1> ;
3194                    <1> ;           THESE PARAMETERS OF DISK BASE MUST BE CHANGED IN ORDER TO
3195                    <1> ;           FORMAT THE FOLLOWING MEDIAS:
3196                    <1> ;           ---------------------------------------------
3197                    <1> ;           : MEDIA  :     DRIVE      : PARM 1 : PARM 2 :
3198                    <1> ;           ---------------------------------------------
3199                    <1> ;           : 320K  : 320K/360K/1.2M :  50H   :   8    :
3200                    <1> ;           : 360K  : 320K/360K/1.2M :  50H   :   9    :
3201                    <1> ;           : 1.2M  : 1.2M           :  54H   :  15    :
3202                    <1> ;           : 720K  : 720K/1.44M     :  50H   :   9    :
3203                    <1> ;           : 1.44M       : 1.44M          :  6CH   :  18    :
3204                    <1> ;           ---------------------------------------------
3205                    <1> ;           NOTES: - PARM 1 = GAP LENGTH FOR FORMAT
3206                    <1> ;                  - PARM 2 = EOT (LAST SECTOR ON TRACK)
3207                    <1> ;                  - DISK BASE IS POINTED BY DISK POINTER LOCATED
3208                    <1> ;                   AT ABSOLUTE ADDRESS 0:78.
3209                    <1> ;                  - WHEN FORMAT OPERATIONS ARE COMPLETE, THE PARAMETERS
3210                    <1> ;                   SHOULD BE RESTORED TO THEIR RESPECTIVE INITIAL VALUES.
3211                    <1> ;-----------------------------------------------------------------------------
3212                    <1> ;     (AH) = 08H READ DRIVE PARAMETERS
3213                    <1> ;     REGISTERS
3214                    <1> ;       INPUT
3215                    <1> ;         (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
3216                    <1> ;          ** 27/05/2016 - TRDOS 386 (TRDOS v2.0) **
3217                    <1> ;            ** EBX = Buffer address for floppy disk parameters table **
3218                    <1> ;       OUTPUT
3219                    <1> ;         (ES:DI) POINTS TO DRIVE PARAMETER TABLE
3220                    <1> ;         *** TRDOS 386 note: floppy disk parameter table (16 bytes)
3221                    <1> ;         will be returned to user in EBX, buffer address *** 27/05/2016 ***
3222                    <1> ;
3223                    <1> ;         (CH) - LOW ORDER 8 OF 10 BITS MAXIMUM NUMBER OF TRACKS
3224                    <1> ;         (CL) - BITS 7 & 6 - HIGH ORDER TWO BITS OF MAXIMUM TRACKS
3225                    <1> ;                BITS 5 THRU 0 - MAXIMUM SECTORS PER TRACK
3226                    <1> ;         (DH) - MAXIMUM HEAD NUMBER
3227                    <1> ;         (DL) - NUMBER OF DISKETTE DRIVES INSTALLED
3228                    <1> ;         (BH) - 0
3229                    <1> ;         (BL) - BITS 7 THRU 4 - 0
3230                    <1> ;                BITS 3 THRU 0 - VALID DRIVE TYPE VALUE IN CMOS
3231                    <1> ;         (AX) - 0
3232                    <1> ;      UNDER THE FOLLOWING CIRCUMSTANCES:
3233                    <1> ;         (1) THE DRIVE NUMBER IS INVALID,
3234                    <1> ;         (2) THE DRIVE TYPE IS UNKNOWN AND CMOS IS NOT PRESENT,
3235                    <1> ;         (3) THE DRIVE TYPE IS UNKNOWN AND CMOS IS BAD,
3236                    <1> ;         (4) OR THE DRIVE TYPE IS UNKNOWN AND THE CMOS DRIVE TYPE IS INVALID
3237                    <1> ;         THEN ES,AX,BX,CX,DH,DI=0 ; DL=NUMBER OF DRIVES.
3238                    <1> ;         IF NO DRIVES ARE PRESENT THEN: ES,AX,BX,CX,DX,DI=0.
3239                    <1> ;         @DISKETTE_STATUS = 0 AND CY IS RESET.
3240                    <1> ;-----------------------------------------------------------------------------
3241                    <1> ;     (AH)= 15H  READ DASD TYPE
3242                    <1> ;     OUTPUT REGISTERS
3243                    <1> ;     (AH) - ON RETURN IF CARRY FLAG NOT SET, OTHERWISE ERROR
3244                    <1> ;           00 - DRIVE NOT PRESENT
3245                    <1> ;           01 - DISKETTE, NO CHANGE LINE AVAILABLE
3246                    <1> ;           02 - DISKETTE, CHANGE LINE AVAILABLE
3247                    <1> ;           03 - RESERVED (FIXED DISK)
3248                    <1> ;     (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
3249                    <1> ;-----------------------------------------------------------------------------
3250                    <1> ;     (AH)= 16H  DISK CHANGE LINE STATUS
3251                    <1> ;     OUTPUT REGISTERS
3252                    <1> ;     (AH) - 00 - DISK CHANGE LINE NOT ACTIVE
3253                    <1> ;            06 - DISK CHANGE LINE ACTIVE & CARRY BIT ON
3254                    <1> ;     (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
3255                    <1> ;-----------------------------------------------------------------------------
3256                    <1> ;     (AH)= 17H  SET DASD TYPE FOR FORMAT
3257                    <1> ;     INPUT REGISTERS
3258                    <1> ;     (AL) - 00 - NOT USED
3259                    <1> ;           01 - DISKETTE 320/360K IN 360K DRIVE
3260                    <1> ;           02 - DISKETTE 360K IN 1.2M DRIVE
3261                    <1> ;           03 - DISKETTE 1.2M IN 1.2M DRIVE
3262                    <1> ;           04 - DISKETTE 720K IN 720K DRIVE
3263                    <1> ;     (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED:
3264                    <1> ;            (DO NOT USE WHEN DISKETTE ATTACH CARD USED)
3265                    <1> ;-----------------------------------------------------------------------------
3266                    <1> ;     (AH)= 18H  SET MEDIA TYPE FOR FORMAT
3267                    <1> ;     INPUT REGISTERS
3268                    <1> ;     (CH) - LOW ORDER 8 OF 10 BITS MAXIMUM TRACKS
3269                    <1> ;     (CL) - BITS 7 & 6 - HIGH ORDER TWO BITS OF MAXIMUM TRACKS
3270                    <1> ;            BITS 5 THRU 0 - MAXIMUM SECTORS PER TRACK
3271                    <1> ;     (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHACKED)
3272                    <1> ;     OUTPUT REGISTERS:
3273                    <1> ;     (ES:DI) - POINTER TO DRIVE PARAMETERS TABLE FOR THIS MEDIA TYPE,
3274                    <1> ;              UNCHANGED IF (AH) IS NON-ZERO
3275                    <1> ;     (AH) - 00H, CY = 0, TRACK AND SECTORS/TRACK COMBINATION IS SUPPORTED
3276                    <1> ;          - 01H, CY = 1, FUNCTION IS NOT AVAILABLE
3277                    <1> ;          - 0CH, CY = 1, TRACK AND SECTORS/TRACK COMBINATION IS NOT SUPPORTED
3278                    <1> ;          - 80H, CY = 1, TIME OUT (DISKETTE NOT PRESENT)
3279                    <1> ;-----------------------------------------------------------------------------
3280                    <1> ;     DISK CHANGE STATUS IS ONLY CHECKED WHEN A MEDIA SPECIFIED IS OTHER
3281                    <1> ;     THAN 360 KB DRIVE. IF THE DISK CHANGE LINE IS FOUND TO BE
3282                    <1> ;     ACTIVE THE FOLLOWING ACTIONS TAKE PLACE:
3283                    <1> ;           ATTEMPT TO RESET DISK CHANGE LINE TO INACTIVE STATE.
3284                    <1> ;           IF ATTEMPT SUCCEEDS SET DASD TYPE FOR FORMAT AND RETURN DISK
```

```
                        <1> ;           CHANGE ERROR CODE
                        <1> ;           IF ATTEMPT FAILS RETURN TIMEOUT ERROR CODE AND SET DASD TYPE
                        <1> ;           TO A PREDETERMINED STATE INDICATING MEDIA TYPE UNKNOWN.
                        <1> ;     IF THE DISK CHANGE LINE IN INACTIVE PERFORM SET DASD TYPE FOR FORMAT.
                        <1> ;
                        <1> ; DATA VARIABLE -- @DISK_POINTER
                        <1> ;     DOUBLE WORD POINTER TO THE CURRENT SET OF DISKETTE PARAMETERS
                        <1> ;--------------------------------------------------------------------------
                        <1> ; OUTPUT FOR ALL FUNCTIONS
                        <1> ;     AH = STATUS OF OPERATION
                        <1> ;           STATUS BITS ARE DEFINED IN THE EQUATES FOR @DISKETTE_STATUS
                        <1> ;           VARIABLE IN THE DATA SEGMENT OF THIS MODULE
                        <1> ;     CY = 0 SUCCESSFUL OPERATION (AH=0 ON RETURN, EXCEPT FOR READ DASD
                        <1> ;           TYPE AH=(15)).
                        <1> ;     CY = 1 FAILED OPERATION (AH HAS ERROR REASON)
                        <1> ;     FOR READ/WRITE/VERIFY
                        <1> ;           DS,BX,DX,CX PRESERVED
                        <1> ;     NOTE: IF AN ERROR IS REPORTED BY THE DISKETTE CODE, THE APPROPRIATE
                        <1> ;           ACTION IS TO RESET THE DISKETTE, THEN RETRY THE OPERATION.
                        <1> ;           ON READ ACCESSES, NO MOTOR START DELAY IS TAKEN, SO THAT
                        <1> ;           THREE RETRIES ARE REQUIRED ON READS TO ENSURE THAT THE
                        <1> ;           PROBLEM IS NOT DUE TO MOTOR START-UP.
                        <1> ;--------------------------------------------------------------------------
                        <1> ;
                        <1> ; DISKETTE STATE MACHINE - ABSOLUTE ADDRESS 40:90 (DRIVE A) & 91 (DRIVE B)
                        <1> ;
                        <1> ;   ---------------------------------------------------------------
                        <1> ;   |       |       |       |       |       |       |       |       |
                        <1> ;   |   7   |   6   |   5   |   4   |   3   |   2   |   1   |   0   |
                        <1> ;   |       |       |       |       |       |       |       |       |
                        <1> ;   ---------------------------------------------------------------
                        <1> ;     |       |       |       |       |       |       |
                        <1> ;     |       |       |       |       |       -----------------
                        <1> ;     |       |       |       |       |             |
                        <1> ;     |       |       |       |     RESERVED        |
                        <1> ;     |       |       |       |                PRESENT STATE
                        <1> ;     |       |       |       |       000: 360K IN 360K DRIVE UNESTABLISHED
                        <1> ;     |       |       |       |       001: 360K IN 1.2M DRIVE UNESTABLISHED
                        <1> ;     |       |       |       |       010: 1.2M IN 1.2M DRIVE UNESTABLISHED
                        <1> ;     |       |       |       |       011: 360K IN 360K DRIVE ESTABLISHED
                        <1> ;     |       |       |       |       100: 360K IN 1.2M DRIVE ESTABLISHED
                        <1> ;     |       |       |       |       101: 1.2M IN 1.2M DRIVE ESTABLISHED
                        <1> ;     |       |       |       |       110: RESERVED
                        <1> ;     |       |       |       |       111: NONE OF THE ABOVE
                        <1> ;     |       |       |       |
                        <1> ;     |       |       |       ------>       MEDIA/DRIVE ESTABLISHED
                        <1> ;     |       |       |
                        <1> ;     |       |       ---------------->     DOUBLE STEPPING REQUIRED (360K IN 1.2M
                        <1> ;     |       |                       DRIVE)
                        <1> ;     |       |
                        <1> ;     --------------------------------->  DATA TRANSFER RATE FOR THIS DRIVE:
                        <1> ;
                        <1> ;                                          00: 500 KBS
                        <1> ;                                          01: 300 KBS
                        <1> ;                                          10: 250 KBS
                        <1> ;                                          11: RESERVED
                        <1> ;
                        <1> ;
                        <1> ;--------------------------------------------------------------------------
                        <1> ; STATE OPERATION STARTED - ABSOLUTE ADDRESS 40:92 (DRIVE A) & 93 (DRIVE B)
                        <1> ;--------------------------------------------------------------------------
                        <1> ; PRESENT CYLINDER NUMBER - ABSOLUTE ADDRESS 40:94 (DRIVE A) & 95 (DRIVE B)
                        <1> ;--------------------------------------------------------------------------
                        <1>
                        <1> struc MD
00000000 ??             <1>       .SPEC1       resb  1      ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
00000001 ??             <1>       .SPEC2       resb  1      ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
00000002 ??             <1>       .OFF_TIM     resb  1      ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
00000003 ??             <1>       .BYT_SEC     resb  1      ; 512 BYTES/SECTOR
00000004 ??             <1>       .SEC_TRK     resb  1      ; EOT (LAST SECTOR ON TRACK)
00000005 ??             <1>       .GAP         resb  1      ; GAP LENGTH
00000006 ??             <1>       .DTL         resb  1      ; DTL
00000007 ??             <1>       .GAP3        resb  1      ; GAP LENGTH FOR FORMAT
00000008 ??             <1>       .FIL_BYT     resb  1      ; FILL BYTE FOR FORMAT
00000009 ??             <1>       .HD_TIM            resb  1      ; HEAD SETTLE TIME (MILLISECONDS)
0000000A ??             <1>       .STR_TIM     resb  1      ; MOTOR START TIME (1/8 SECONDS)
0000000B ??             <1>       .MAX_TRK     resb  1      ; MAX. TRACK NUMBER
0000000C ??             <1>       .RATE        resb  1      ; DATA TRANSFER RATE
                        <1> endstruc
                        <1>
                        <1> BIT7OFF     EQU   7FH
                        <1> BIT7ON      EQU   80H
                        <1>
                        <1> ; 30/08/2020 - TRDOS 386 v2
                        <1>
                        <1> ;;int13h: ; 16/02/2015
                        <1> ;; 16/02/2015 - 21/02/2015
                        <1> int40h:
                        <1> ; 11/04/2021
                        <1> diskette_io:
000042AE F8             <1>       clc ; 20/07/2020
000042AF 9C             <1>       pushfd
000042B0 0E             <1>       push  cs
000042B1 E801000000     <1>       call  DISKETTE_IO_1
000042B6 C3             <1>       retn
                        <1>
                        <1> DISKETTE_IO_1:
                        <1>
000042B7 FB             <1>       STI                     ; INTERRUPTS BACK ON
000042B8 55             <1>       PUSH  eBP               ; USER REGISTER
000042B9 57             <1>       PUSH  eDI               ; USER REGISTER
000042BA 52             <1>       PUSH  eDX               ; HEAD #, DRIVE # OR USER REGISTER
000042BB 53             <1>       PUSH  eBX               ; BUFFER OFFSET PARAMETER OR REGISTER
000042BC 51             <1>       PUSH  eCX               ; TRACK #-SECTOR # OR USER REGISTER
000042BD 89E5           <1>       MOV   eBP,eSP           ; BP     => PARAMETER LIST DEP. ON AH
```

```
                                <1>                                     ; [BP]   = SECTOR #
                                <1>                                     ; [BP+1] = TRACK #
                                <1>                                     ; [BP+2] = BUFFER OFFSET
                                <1>                                     ; FOR RETURN OF DRIVE PARAMETERS:
                                <1>                                     ; CL/[BP] = BITS 7&6 HI BITS OF MAX CYL
                                <1>                                     ;          BITS 0-5 MAX SECTORS/TRACK
                                <1>                                     ; CH/[BP+1] = LOW 8 BITS OF MAX CYL.
                                <1>                                     ; BL/[BP+2] = BITS 7-4 = 0
                                <1>                                     ;           BITS 3-0 = VALID CMOS TYPE
                                <1>                                     ; BH/[BP+3] = 0
                                <1>                                     ; DL/[BP+4] = # DRIVES INSTALLED
                                <1>                                     ; DH/[BP+5] = MAX HEAD #
                                <1>                                     ; DI/[BP+6] = OFFSET TO DISK BASE
000042BF 06                     <1>       push   es ; 06/02/2015
000042C0 1E                     <1>       PUSH   DS                     ; BUFFER SEGMENT PARM OR USER REGISTER
000042C1 56                     <1>       PUSH   eSI                    ; USER REGISTERS
                                <1>       ;CALL  DDS                    ; SEGMENT OF BIOS DATA AREA TO DS
                                <1>       ;mov   cx, cs
                                <1>       ;mov   ds, cx
000042C2 66B91000               <1>       mov    cx, KDATA
000042C6 8ED9                   <1>       mov   ds, cx
000042C8 8EC1                   <1>       mov   es, cx
                                <1>
                                <1>       ;CMP   AH,(FNC_TAE-FNC_TAB)/2 ; CHECK FOR > LARGEST FUNCTION
000042CA 80FC19                 <1>       cmp    ah,(FNC_TAE-FNC_TAB)/4 ; 18/02/2015
000042CD 7202                   <1>       JB     short OK_FUNC          ; FUNCTION OK
000042CF B414                   <1>       MOV    AH,14H                 ; REPLACE WITH KNOWN INVALID FUNCTION
                                <1> OK_FUNC:
000042D1 80FC01                 <1>       CMP    AH,1                   ; RESET OR STATUS ?
000042D4 760C                   <1>       JBE    short OK_DRV           ; IF RESET OR STATUS DRIVE ALWAYS OK
000042D6 80FC08                 <1>       CMP    AH,8                   ; READ DRIVE PARMS ?
000042D9 7407                   <1>       JZ     short OK_DRV           ; IF SO DRIVE CHECKED LATER
000042DB 80FA01                 <1>       CMP    DL,1                   ; DRIVES 0 AND 1 OK
000042DE 7602                   <1>       JBE    short OK_DRV           ; IF 0 OR 1 THEN JUMP
000042E0 B414                   <1>       MOV    AH,14H                 ; REPLACE WITH KNOWN INVALID FUNCTION
                                <1> OK_DRV:
000042E2 31C9                   <1>       xor    ecx, ecx
                                <1>       ;mov   esi, ecx ; 08/02/2015
000042E4 89CF                   <1>       mov    edi, ecx ; 08/02/2015
000042E6 88E1                   <1>       MOV    CL,AH                  ; CL = FUNCTION
                                <1>       ;XOR   CH,CH                  ; CX = FUNCTION
                                <1>       ;SHL   CL, 1                  ; FUNCTION TIMES 2
000042E8 C0E102                 <1>       SHL    CL, 2 ; 20/02/2015     ; FUNCTION TIMES 4 (for 32 bit offset)
000042EB BB[23430000]           <1>       MOV    eBX,FNC_TAB            ; LOAD START OF FUNCTION TABLE
000042F0 01CB                   <1>       ADD    eBX,eCX                     ; ADD OFFSET INTO TABLE => ROUTINE
000042F2 88F4                   <1>       MOV    AH,DH                  ; AX = HEAD #,# OF SECTORS OR DASD TYPE
000042F4 30F6                   <1>       XOR    DH,DH                  ; DX = DRIVE #
000042F6 6689C6                 <1>       MOV    SI,AX                  ; SI = HEAD #,# OF SECTORS OR DASD TYPE
000042F9 6689D7                 <1>       MOV     DI,DX                 ; DI = DRIVE #
                                <1>       ;
                                <1>       ; 11/12/2014
000042FC 8815[DD680000]         <1>       mov   [cfd], dl               ; current floppy drive (for 'GET_PARM')
                                <1>       ;
00004302 8A25[30810100]         <1>       MOV    AH, [DSKETTE_STATUS]    ; LOAD STATUS TO AH FOR STATUS FUNCTION
00004308 C605[30810100]00       <1>       MOV    byte [DSKETTE_STATUS],0 ; INITIALIZE FOR ALL OTHERS
                                <1>
                                <1> ;     THROUGHOUT THE DISKETTE BIOS, THE FOLLOWING INFORMATION IS CONTAINED IN
                                <1> ;     THE FOLLOWING MEMORY LOCATIONS AND REGISTERS. NOT ALL DISKETTE BIOS
                                <1> ;     FUNCTIONS REQUIRE ALL OF THESE PARAMETERS.
                                <1> ;
                                <1> ;           DI     : DRIVE #
                                <1> ;           SI-HI  : HEAD #
                                <1> ;           SI-LOW : # OF SECTORS OR DASD TYPE FOR FORMAT
                                <1> ;           ES     : BUFFER SEGMENT
                                <1> ;           [BP]   : SECTOR #
                                <1> ;           [BP+1] : TRACK #
                                <1> ;           [BP+2] : BUFFER OFFSET
                                <1> ;
                                <1> ;     ACROSS CALLS TO SUBROUTINES THE CARRY FLAG (CY=1), WHERE INDICATED IN
                                <1> ;     SUBROUTINE PROLOGUES, REPRESENTS AN EXCEPTION RETURN (NORMALLY AN ERROR
                                <1> ;     CONDITION). IN MOST CASES, WHEN CY = 1, @DSKETTE_STATUS CONTAINS THE
                                <1> ;     SPECIFIC ERROR CODE.
                                <1> ;
                                <1>                                     ; (AH) = @DSKETTE_STATUS
0000430F FF13                   <1>       CALL   dWORD [eBX]            ; CALL THE REQUESTED FUNCTION
00004311 5E                     <1>       POP    eSI                    ; RESTORE ALL REGISTERS
00004312 1F                     <1>       POP    DS
00004313 07                     <1>       pop    es     ; 06/02/2015
00004314 59                     <1>       POP    eCX
00004315 5B                     <1>       POP    eBX
00004316 5A                     <1>       POP    eDX
00004317 5F                     <1>       POP    eDI
00004318 89E5                   <1>       MOV    eBP, eSP
0000431A 50                     <1>       PUSH   eAX
0000431B 9C                     <1>       PUSHFd
0000431C 58                     <1>       POP    eAX
                                <1>       ;MOV   [BP+6], AX
0000431D 89450C                 <1>       mov    [ebp+12], eax  ; 18/02/2015, flags
00004320 58                     <1>       POP    eAX
00004321 5D                     <1>       POP    eBP
00004322 CF                     <1>       IRETd
                                <1>
                                <1> ;--------------------------------------------------------------------------
                                <1> ; DW --> dd (06/02/2015)
00004323 [87430000]             <1> FNC_TAB dd  DSK_RESET             ; AH = 00H; RESET
00004327 [FD430000]             <1>       dd    DSK_STATUS            ; AH = 01H; STATUS
0000432B [0E440000]             <1>       dd    DSK_READ              ; AH = 02H; READ
0000432F [1F440000]             <1>       dd    DSK_WRITE             ; AH = 03H; WRITE
00004333 [30440000]             <1>       dd    DSK_VERF              ; AH = 04H; VERIFY
00004337 [41440000]             <1>       dd    DSK_FORMAT            ; AH = 05H; FORMAT
0000433B [C6440000]             <1>       dd    FNC_ERR               ; AH = 06H; INVALID
0000433F [C6440000]             <1>       dd    FNC_ERR               ; AH = 07H; INVALID
00004343 [D3440000]             <1>       dd    DSK_PARMS             ; AH = 08H; READ DRIVE PARAMETERS
00004347 [C6440000]             <1>       dd    FNC_ERR               ; AH = 09H; INVALID
0000434B [C6440000]             <1>       dd    FNC_ERR               ; AH = 0AH; INVALID
```

```
3495 0000434F [C6440000]          <1>       dd     FNC_ERR             ; AH = 0BH; INVALID
3496 00004353 [C6440000]          <1>       dd     FNC_ERR             ; AH = 0CH; INVALID
3497 00004357 [C6440000]          <1>       dd     FNC_ERR             ; AH = 0DH; INVALID
3498 0000435B [C6440000]          <1>       dd     FNC_ERR             ; AH = 0EH; INVALID
3499 0000435F [C6440000]          <1>       dd     FNC_ERR             ; AH = 0FH; INVALID
3500 00004363 [C6440000]          <1>       dd     FNC_ERR             ; AH = 10H; INVALID
3501 00004367 [C6440000]          <1>       dd     FNC_ERR             ; AH = 11H; INVALID
3502 0000436B [C6440000]          <1>       dd     FNC_ERR             ; AH = 12H; INVALID
3503 0000436F [C6440000]          <1>       dd     FNC_ERR             ; AH = 13H; INVALID
3504 00004373 [C6440000]          <1>       dd     FNC_ERR             ; AH = 14H; INVALID
3505 00004377 [B9450000]          <1>       dd     DSK_TYPE            ; AH = 15H; READ DASD TYPE
3506 0000437B [E7450000]          <1>       dd     DSK_CHANGE          ; AH = 16H; CHANGE STATUS
3507 0000437F [21460000]          <1>       dd     FORMAT_SET          ; AH = 17H; SET DASD TYPE
3508 00004383 [9B460000]          <1>       dd     SET_MEDIA           ; AH = 18H; SET MEDIA TYPE
3509                              <1> FNC_TAE EQU  $                   ; END
3510                              <1>
3511                              <1> ;--------------------------------------------------------------------------
3512                              <1> ; DISK_RESET (AH = 00H)
3513                              <1> ;           RESET THE DISKETTE SYSTEM.
3514                              <1> ;
3515                              <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
3516                              <1> ;--------------------------------------------------------------------------
3517                              <1> DSK_RESET:
3518 00004387 66BAF203            <1>       MOV    DX,03F2H             ; ADAPTER CONTROL PORT
3519 0000438B FA                  <1>       CLI                         ; NO INTERRUPTS
3520 0000438C A0[2E810100]        <1>       MOV    AL,[MOTOR_STATUS]    ; GET DIGITAL OUTPUT REGISTER REFLECTION
3521 00004391 243F                <1>       AND    AL,00111111B         ; KEEP SELECTED AND MOTOR ON BITS
3522 00004393 C0C004              <1>       ROL    AL,4                 ; MOTOR VALUE TO HIGH NIBBLE
3523                              <1>                                   ; DRIVE SELECT TO LOW NIBBLE
3524 00004396 0C08                <1>       OR     AL,00001000B         ; TURN ON INTERRUPT ENABLE
3525 00004398 EE                  <1>       OUT    DX,AL                ; RESET THE ADAPTER
3526 00004399 C605[2D810100]00    <1>       MOV    byte [SEEK_STATUS],0 ; SET RECALIBRATE REQUIRED ON ALL DRIVES
3527                              <1>       ;JMP   $+2                  ; WAIT FOR I/O
3528                              <1>       ;JMP   $+2                  ; WAIT FOR I/O (TO INSURE MINIMUM
3529                              <1>                                   ;  PULSE WIDTH)
3530                              <1>       ; 19/12/2014
3531                              <1>       NEWIODELAY
2995 000043A0 E6EB                <2>  out 0ebh,al
3532                              <1>
3533                              <1>       ; 17/12/2014
3534                              <1>       ; AWARD BIOS 1999 - RESETDRIVES (ADISK.ASM)
3535 000043A2 B915000000          <1>       mov    ecx, WAITCPU_RESET_ON     ; cx = 21 -- Min. 14 micro seconds !?
3536                              <1> wdw1:
3537                              <1>       NEWIODELAY   ; 27/02/2015
2995 000043A7 E6EB                <2>  out 0ebh,al
3538 000043A9 E2FC                <1>       loop   wdw1
3539                              <1>       ;
3540 000043AB 0C04                <1>       OR     AL,00000100B         ; TURN OFF RESET BIT
3541 000043AD EE                  <1>       OUT    DX,AL                ; RESET THE ADAPTER
3542                              <1>       ; 16/12/2014
3543                              <1>       IODELAY
2990 000043AE EB00                <2>  jmp short $+2
2991 000043B0 EB00                <2>  jmp short $+2
3544                              <1>       ;
3545                              <1>       ;STI                        ; ENABLE THE INTERRUPTS
3546 000043B2 E8210C0000          <1>       CALL   WAIT_INT             ; WAIT FOR THE INTERRUPT
3547 000043B7 723B                <1>       JC     short DR_ERR         ; IF ERROR, RETURN IT
3548 000043B9 66B9C000            <1>       MOV    CX,11000000B         ; CL = EXPECTED @NEC_STATUS
3549                              <1> NXT_DRV:
3550                              <1>       ;PUSH  CX                   ; SAVE FOR CALL
3551                              <1>       ; 11/04/2021
3552 000043BD 51                  <1>       push   ecx
3553 000043BE B8[F3430000]        <1>       MOV    eAX, DR_POP_ERR      ; LOAD NEC_OUTPUT ERROR ADDRESS
3554 000043C3 50                  <1>       PUSH   eAX                  ; "
3555 000043C4 B408                <1>       MOV    AH,08H               ; SENSE INTERRUPT STATUS COMMAND
3556 000043C6 E8020B0000          <1>       CALL   NEC_OUTPUT
3557 000043CB 58                  <1>       POP    eAX                  ; THROW AWAY ERROR RETURN
3558 000043CC E8370C0000          <1>       CALL   RESULTS                    ; READ IN THE RESULTS
3559                              <1>       ;POP   CX                   ; RESTORE AFTER CALL
3560                              <1>       ; 11/04/2021
3561 000043D1 59                  <1>       pop    ecx
3562 000043D2 7220                <1>       JC     short DR_ERR         ; ERROR RETURN
3563 000043D4 3A0D[31810100]      <1>       CMP    CL, [NEC_STATUS]     ; TEST FOR DRIVE READY TRANSITION
3564 000043DA 7518                <1>       JNZ    short DR_ERR         ; EVERYTHING OK
3565 000043DC FEC1                <1>       INC    CL                   ; NEXT EXPECTED @NEC_STATUS
3566 000043DE 80F9C3              <1>       CMP    CL,11000011B         ; ALL POSSIBLE DRIVES CLEARED
3567 000043E1 76DA                <1>       JBE    short NXT_DRV        ; FALL THRU IF 11000100B OR >
3568                              <1>       ;
3569 000043E3 E86E030000          <1>       CALL   SEND_SPEC            ; SEND SPECIFY COMMAND TO NEC
3570                              <1> RESBAC:
3571 000043E8 E80C090000          <1>       CALL   SETUP_END            ; VARIOUS CLEANUPS
3572 000043ED 6689F3              <1>       MOV    BX,SI                ; GET SAVED AL TO BL
3573 000043F0 88D8                <1>       MOV    AL,BL                ; PUT BACK FOR RETURN
3574 000043F2 C3                  <1>       RETn
3575                              <1> DR_POP_ERR:
3576                              <1>       ;POP   CX                   ; CLEAR STACK
3577                              <1>       ; 11/04/2021
3578 000043F3 59                  <1>       pop    ecx
3579                              <1> DR_ERR:
3580 000043F4 800D[30810100]20    <1>       OR     byte [DSKETTE_STATUS],BAD_NEC ; SET ERROR CODE
3581 000043FB EBEB                <1>       JMP    SHORT RESBAC         ; RETURN FROM RESET
3582                              <1>
3583                              <1> ;--------------------------------------------------------------------------
3584                              <1> ; DISK_STATUS      (AH = 01H)
3585                              <1> ;     DISKETTE STATUS.
3586                              <1> ;
3587                              <1> ; ON ENTRY: AH : STATUS OF PREVIOUS OPERATION
3588                              <1> ;
3589                              <1> ; ON EXIT:  AH, @DSKETTE_STATUS, CY REFLECT STATUS OF PREVIOUS OPERATION.
3590                              <1> ;--------------------------------------------------------------------------
3591                              <1> DSK_STATUS:
3592 000043FD 8825[30810100]      <1>       MOV    [DSKETTE_STATUS],AH ; PUT BACK FOR SETUP END
3593 00004403 E8F1080000          <1>       CALL   SETUP_END            ; VARIOUS CLEANUPS
3594 00004408 6689F3              <1>       MOV    BX,SI                ; GET SAVED AL TO BL
3595 0000440B 88D8                <1>       MOV    AL,BL                ; PUT BACK FOR RETURN
```

```
3596 0000440D C3                  <1>       RETn
3597                              <1>
3598                              <1> ;--------------------------------------------------------------------------------
3599                              <1> ; DISK_READ  (AH = 02H)
3600                              <1> ;     DISKETTE READ.
3601                              <1> ;
3602                              <1> ; ON ENTRY:  DI     : DRIVE #
3603                              <1> ;            SI-HI  : HEAD #
3604                              <1> ;            SI-LOW : # OF SECTORS
3605                              <1> ;            ES     : BUFFER SEGMENT
3606                              <1> ;            [BP]   : SECTOR #
3607                              <1> ;            [BP+1] : TRACK #
3608                              <1> ;            [BP+2] : BUFFER OFFSET
3609                              <1> ;
3610                              <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
3611                              <1> ;--------------------------------------------------------------------------------
3612                              <1>
3613                              <1> ; 06/02/2015, ES:BX -> EBX (unix386.s)
3614                              <1>
3615                              <1> DSK_READ:
3616 0000440E 8025[2E810100]7F    <1>       AND    byte [MOTOR_STATUS],01111111B ; INDICATE A READ OPERATION
3617 00004415 66B846E6            <1>       MOV    AX,0E646H           ; AX = NEC COMMAND, DMA COMMAND
3618 00004419 E842040000          <1>       CALL   RD_WR_VF            ; COMMON READ/WRITE/VERIFY
3619 0000441E C3                  <1>       RETn
3620                              <1>
3621                              <1> ;--------------------------------------------------------------------------------
3622                              <1> ; DISK_WRITE (AH = 03H)
3623                              <1> ;     DISKETTE WRITE.
3624                              <1> ;
3625                              <1> ; ON ENTRY:  DI     : DRIVE #
3626                              <1> ;            SI-HI  : HEAD #
3627                              <1> ;            SI-LOW : # OF SECTORS
3628                              <1> ;            ES     : BUFFER SEGMENT
3629                              <1> ;            [BP]   : SECTOR #
3630                              <1> ;            [BP+1] : TRACK #
3631                              <1> ;            [BP+2] : BUFFER OFFSET
3632                              <1> ;
3633                              <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
3634                              <1> ;--------------------------------------------------------------------------------
3635                              <1>
3636                              <1> ; 06/02/2015, ES:BX -> EBX (unix386.s)
3637                              <1>
3638                              <1> DSK_WRITE:
3639 0000441F 66B84AC5            <1>       MOV    AX,0C54AH           ; AX = NEC COMMAND, DMA COMMAND
3640 00004423 800D[2E810100]80    <1>       OR    byte [MOTOR_STATUS],10000000B ; INDICATE WRITE OPERATION
3641 0000442A E831040000          <1>       CALL   RD_WR_VF            ; COMMON READ/WRITE/VERIFY
3642 0000442F C3                  <1>       RETn
3643                              <1>
3644                              <1> ;--------------------------------------------------------------------------------
3645                              <1> ; DISK_VERF  (AH = 04H)
3646                              <1> ;     DISKETTE VERIFY.
3647                              <1> ;
3648                              <1> ; ON ENTRY:  DI     : DRIVE #
3649                              <1> ;            SI-HI  : HEAD #
3650                              <1> ;            SI-LOW : # OF SECTORS
3651                              <1> ;            ES     : BUFFER SEGMENT
3652                              <1> ;            [BP]   : SECTOR #
3653                              <1> ;            [BP+1] : TRACK #
3654                              <1> ;            [BP+2] : BUFFER OFFSET
3655                              <1> ;
3656                              <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
3657                              <1> ;--------------------------------------------------------------------------------
3658                              <1> DSK_VERF:
3659 00004430 8025[2E810100]7F    <1>       AND    byte [MOTOR_STATUS],01111111B ; INDICATE A READ OPERATION
3660 00004437 66B842E6            <1>       MOV    AX,0E642H           ; AX = NEC COMMAND, DMA COMMAND
3661 0000443B E820040000          <1>       CALL   RD_WR_VF            ; COMMON READ/WRITE/VERIFY
3662 00004440 C3                  <1>       RETn
3663                              <1>
3664                              <1> ;--------------------------------------------------------------------------------
3665                              <1> ; DISK_FORMAT      (AH = 05H)
3666                              <1> ;     DISKETTE FORMAT.
3667                              <1> ;
3668                              <1> ; ON ENTRY:  DI     : DRIVE #
3669                              <1> ;            SI-HI  : HEAD #
3670                              <1> ;            SI-LOW : # OF SECTORS
3671                              <1> ;            ES     : BUFFER SEGMENT
3672                              <1> ;            [BP]   : SECTOR #
3673                              <1> ;            [BP+1] : TRACK #
3674                              <1> ;            [BP+2] : BUFFER OFFSET
3675                              <1> ;            @DISK_POINTER POINTS TO THE PARAMETER TABLE OF THIS DRIVE
3676                              <1> ;
3677                              <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
3678                              <1> ;--------------------------------------------------------------------------------
3679                              <1> DSK_FORMAT:
3680 00004441 E859030000          <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
3681 00004446 E849050000          <1>       CALL   FMT_INIT            ; ESTABLISH STATE IF UNESTABLISHED
3682 0000444B 800D[2E810100]80    <1>       OR    byte [MOTOR_STATUS], 10000000B ; INDICATE WRITE OPERATION
3683 00004452 E891050000          <1>       CALL   MED_CHANGE          ; CHECK MEDIA CHANGE AND RESET IF SO
3684 00004457 725D                <1>       JC    short FM_DON         ; MEDIA CHANGED, SKIP
3685 00004459 E8F8020000          <1>       CALL   SEND_SPEC           ; SEND SPECIFY COMMAND TO NEC
3686 0000445E E8F5050000          <1>       CALL   CHK_LASTRATE        ; ZF=1 ATTEMPT RATE IS SAME AS LAST RATE
3687 00004463 7405                <1>       JZ    short FM_WR          ; YES, SKIP SPECIFY COMMAND
3688 00004465 E8CE050000          <1>       CALL   SEND_RATE           ; SEND DATA RATE TO CONTROLLER
3689                              <1> FM_WR:
3690 0000446A E87E060000          <1>       CALL   FMTDMA_SET          ; SET UP THE DMA FOR FORMAT
3691 0000446F 7245                <1>       JC     short FM_DON        ; RETURN WITH ERROR
3692 00004471 B44D                <1>       MOV    AH,04DH             ; ESTABLISH THE FORMAT COMMAND
3693 00004473 E8D9060000          <1>       CALL   NEC_INIT            ; INITIALIZE THE NEC
3694 00004478 723C                <1>       JC    short FM_DON         ; ERROR - EXIT
3695 0000447A B8[B6440000]        <1>       MOV   eAX, FM_DON          ; LOAD ERROR ADDRESS
3696 0000447F 50                  <1>       PUSH   eAX                 ; PUSH NEC_OUT ERROR RETURN
3697 00004480 B203                <1>       MOV    DL,3                ; BYTES/SECTOR VALUE TO NEC
3698 00004482 E840090000          <1>       CALL   GET_PARM
3699 00004487 E8410A0000          <1>       CALL   NEC_OUTPUT
3700 0000448C B204                <1>       MOV    DL,4                ; SECTORS/TRACK VALUE TO NEC
```

```
3701 0000448E E834090000          <1>       CALL   GET_PARM
3702 00004493 E8350A0000          <1>       CALL   NEC_OUTPUT
3703 00004498 B207                <1>       MOV    DL,7                ; GAP LENGTH VALUE TO NEC
3704 0000449A E828090000          <1>       CALL   GET_PARM
3705 0000449F E8290A0000          <1>       CALL   NEC_OUTPUT
3706 000044A4 B208                <1>       MOV    DL,8                ; FILLER BYTE TO NEC
3707 000044A6 E81C090000          <1>       CALL   GET_PARM
3708 000044AB E81D0A0000          <1>       CALL   NEC_OUTPUT
3709 000044B0 58                  <1>       POP    eAX                 ; THROW AWAY ERROR
3710 000044B1 E817070000          <1>       CALL   NEC_TERM            ; TERMINATE, RECEIVE STATUS, ETC,
3711                               <1> FM_DON:
3712 000044B6 E815030000          <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
3713 000044BB E839080000          <1>       CALL   SETUP_END           ; VARIOUS CLEANUPS
3714 000044C0 6689F3              <1>       MOV    BX,SI               ; GET SAVED AL TO BL
3715 000044C3 88D8                <1>       MOV    AL,BL               ; PUT BACK FOR RETURN
3716 000044C5 C3                  <1>       RETn
3717                               <1>
3718                               <1> ;--------------------------------------------------------------------------
3719                               <1> ; FNC_ERR
3720                               <1> ;     INVALID FUNCTION REQUESTED OR INVALID DRIVE:
3721                               <1> ;     SET BAD COMMAND IN STATUS.
3722                               <1> ;
3723                               <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
3724                               <1> ;--------------------------------------------------------------------------
3725                               <1> FNC_ERR:                           ; INVALID FUNCTION REQUEST
3726 000044C6 6689F0              <1>       MOV    AX,SI               ; RESTORE AL
3727 000044C9 B401                <1>       MOV    AH,BAD_CMD          ; SET BAD COMMAND ERROR
3728 000044CB 8825[30810100]      <1>       MOV    [DSKETTE_STATUS],AH ; STORE IN DATA AREA
3729 000044D1 F9                  <1>       STC                        ; SET CARRY INDICATING ERROR
3730 000044D2 C3                  <1>       RETn
3731                               <1>
3732                               <1> ; 30/08/2020
3733                               <1> ; 29/08/2020
3734                               <1> ; 01/06/2016
3735                               <1> ; 28/05/2016
3736                               <1> ; 27/05/2016 - TRDOS 386 (TRDOS v.2.0)
3737                               <1> ;--------------------------------------------------------------------------
3738                               <1> ; DISK_PARMS (AH = 08H)
3739                               <1> ;     READ DRIVE PARAMETERS.
3740                               <1> ;
3741                               <1> ; ON ENTRY:  DI : DRIVE #
3742                               <1> ;            ; 27/05/2016
3743                               <1> ;            EBX = Buffer Address for floppy disk parameters table (16 bytes)
3744                               <1> ;
3745                               <1> ; ON EXIT:   CL/[BP]   = BITS 7 & 6 HI 2 BITS OF MAX CYLINDER
3746                               <1> ;                        BITS 0-5 MAX SECTORS/TRACK
3747                               <1> ;            CH/[BP+1] = LOW 8 BITS OF MAX CYLINDER
3748                               <1> ;            BL/[BP+2] = BITS 7-4 = 0
3749                               <1> ;                        BITS 3-0 = VALID CMOS DRIVE TYPE
3750                               <1> ;            BH/[BP+3] = 0
3751                               <1> ;            DL/[BP+4] = # DRIVES INSTALLED (VALUE CHECKED)
3752                               <1> ;            DH/[BP+5] = MAX HEAD #
3753                               <1> ;          ** 27/05/2016 - TRDOS 386 (TRDOS v2.0) **
3754                               <1> ;            ** EBX = Buffer address for floppy disk parameters table **
3755                               <1> ;            ;DI/[BP+6] = OFFSET TO DISK_BASE
3756                               <1> ;            ;ES        = SEGMENT OF DISK_BASE
3757                               <1> ;
3758                               <1> ;            AX        = 0
3759                               <1> ;
3760                               <1> ;            NOTE : THE ABOVE INFORMATION IS STORED IN THE USERS STACK AT
3761                               <1> ;                   THE LOCATIONS WHERE THE MAIN ROUTINE WILL POP THEM
3762                               <1> ;                   INTO THE APPROPRIATE REGISTERS BEFORE RETURNING TO THE
3763                               <1> ;                   CALLER.
3764                               <1> ;--------------------------------------------------------------------------
3765                               <1> DSK_PARMS:
3766 000044D3 E8C7020000          <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH,
3767                               <1>      ;  MOV   WORD [BP+2],0      ; DRIVE TYPE = 0
3768                               <1>      ;  MOV   AX, [EQUIP_FLAG]   ; LOAD EQUIPMENT FLAG FOR # DISKETTES
3769                               <1>      ;  AND   AL,11000001B       ; KEEP DISKETTE DRIVE BITS
3770                               <1>      ;  MOV   DL,2               ; DISKETTE DRIVES = 2
3771                               <1>      ;  CMP   AL,01000001B       ; 2 DRIVES INSTALLED ?
3772                               <1>      ;  JZ    short STO_DL       ; IF YES JUMP
3773                               <1>      ;  DEC   DL                 ; DISKETTE DRIVES = 1
3774                               <1>      ;  CMP   AL,00000001B       ; 1 DRIVE INSTALLED ?
3775                               <1>      ;  JNZ   short NON_DRV      ; IF NO JUMP
3776 000044D8 29D2                <1>       sub    edx, edx
3777 000044DA 66A1[EE680000]      <1>       mov    ax, [fd0_type]
3778 000044E0 6621C0              <1>       and    ax, ax
3779 000044E3 0F8491000000        <1>       jz     NON_DRV
3780 000044E9 FEC2                <1>       inc    dl
3781 000044EB 20E4                <1>       and    ah, ah
3782 000044ED 7402                <1>       jz     short STO_DL
3783 000044EF FEC2                <1>       inc    dl
3784                               <1> STO_DL:
3785                               <1>       ; 30/08/2020
3786 000044F1 6639FA              <1>       cmp    dx, di
3787 000044F4 0F8680000000        <1>       jna    NON_DRV
3788                               <1>       ;
3789                               <1>       ;MOV   [BP+4],DL           ; STORE NUMBER OF DRIVES
3790 000044FA 895508              <1>       mov    [ebp+8], edx ; 20/02/2015
3791                               <1>       ; 11/04/2021
3792                               <1>       ;CMP   DI,1                ; CHECK FOR VALID DRIVE
3793                               <1>       ;;JA   short NON_DRV1      ; DRIVE INVALID
3794                               <1>       ;ja    NON_DRV1 ; 29/08/2020
3795                               <1>       ;
3796                               <1>       ;MOV   BYTE [BP+5],1       ; MAXIMUM HEAD NUMBER =   1
3797 000044FD C6450901            <1>       mov    byte [ebp+9], 1  ; 20/02/2015
3798 00004501 E8B8080000          <1>       CALL   CMOS_TYPE           ; RETURN DRIVE TYPE IN AL
3799                               <1>       ;;20/02/2015
3800                               <1>       ;;JC   short CHK_EST       ; IF CMOS BAD CHECKSUM ESTABLISHED
3801                               <1>       ;;OR   AL,AL               ; TEST FOR NO DRIVE TYPE
3802 00004506 740F                <1>       JZ     short CHK_EST       ; JUMP IF SO
3803 00004508 E820020000          <1>       CALL   DR_TYPE_CHECK       ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
3804 0000450D 7208                <1>       JC     short CHK_EST       ; TYPE NOT IN TABLE (POSSIBLE BAD CMOS)
3805                               <1>       ;MOV   [BP+2],AL           ; STORE VALID CMOS DRIVE TYPE
```

```
3806                                    <1>       ;mov   [ebp+4], al ; 06/02/2015
3807 0000450F 8A4B04                    <1>       MOV   CL, [eBX+MD.SEC_TRK]  ; GET SECTOR/TRACK
3808 00004512 8A6B0B                    <1>       MOV   CH, [eBX+MD.MAX_TRK]  ; GET MAX. TRACK NUMBER
3809 00004515 EB36                      <1>       JMP    SHORT STO_CX        ; CMOS GOOD, USE CMOS
3810                                    <1> CHK_EST:
3811 00004517 8AA7[3D810100]            <1>       MOV    AH, [DSK_STATE+eDI] ; LOAD STATE FOR THIS DRIVE
3812 0000451D F6C410                    <1>       TEST   AH,MED_DET          ; CHECK FOR ESTABLISHED STATE
3813 00004520 745B                      <1>       JZ     short NON_DRV1      ; CMOS BAD/INVALID OR UNESTABLISHED
3814                                    <1> USE_EST:
3815 00004522 80E4C0                    <1>       AND    AH,RATE_MSK         ; ISOLATE STATE
3816 00004525 80FC80                    <1>       CMP    AH,RATE_250         ; RATE 250 ?
3817 00004528 757B                      <1>       JNE    short USE_EST2      ; NO, GO CHECK OTHER RATE
3818                                    <1>
3819                                    <1> ;----- DATA RATE IS 250 KBS, TRY 360 KB TABLE FIRST
3820                                    <1>
3821 0000452A B001                      <1>       MOV    AL,01               ; DRIVE TYPE 1 (360KB)
3822 0000452C E8FC010000                <1>       CALL   DR_TYPE_CHECK       ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
3823 00004531 8A4B04                    <1>       MOV   CL, [eBX+MD.SEC_TRK] ; GET SECTOR/TRACK
3824 00004534 8A6B0B                    <1>       MOV   CH, [eBX+MD.MAX_TRK] ; GET MAX. TRACK NUMBER
3825 00004537 F687[3D810100]01          <1>       TEST   byte [DSK_STATE+eDI],TRK_CAPA ; 80 TRACK ?
3826 0000453E 740D                      <1>       JZ     short STO_CX        ; MUST BE 360KB DRIVE
3827                                    <1>
3828                                    <1> ;----- IT IS 1.44 MB DRIVE
3829                                    <1>
3830                                    <1> PARM144:
3831 00004540 B004                      <1>       MOV    AL,04               ; DRIVE TYPE 4 (1.44MB)
3832 00004542 E8E6010000                <1>       CALL   DR_TYPE_CHECK       ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
3833 00004547 8A4B04                    <1>       MOV   CL, [eBX+MD.SEC_TRK] ; GET SECTOR/TRACK
3834 0000454A 8A6B0B                    <1>       MOV   CH, [eBX+MD.MAX_TRK] ; GET MAX. TRACK NUMBER
3835                                    <1> STO_CX:
3836 0000454D 894D00                    <1>       MOV    [eBP],eCX           ; SAVE POINTER IN STACK FOR RETURN
3837                                    <1> ES_DI:
3838                                    <1>       ;MOV   [BP+6],BX           ; ADDRESS OF MEDIA/DRIVE PARM TABLE
3839                                    <1>       ;mov   [ebp+12], ebx ; 06/02/2015
3840                                    <1>       ;MOV   AX,CS               ; SEGMENT MEDIA/DRIVE PARAMETER TABLE
3841                                    <1>       ;MOV   ES,AX               ; ES IS SEGMENT OF TABLE
3842                                    <1>       ;
3843                                    <1>       ; 28/05/2016
3844                                    <1>       ; 27/05/2016
3845                                    <1>       ; return floppy disk parameters table to user
3846                                    <1>       ; in user's buffer, which is pointed by EBX
3847                                    <1>       ;
3848 00004550 57                       <1>       push   edi
3849 00004551 8B7D04                    <1>       mov    edi, [ebp+4]        ; ebx (input), user's buffer address
3850                                    <1>       ; 29/08/2020
3851 00004554 09FF                      <1>       or     edi, edi
3852 00004556 7417                      <1>       jz     short no_copy_fdpt
3853                                    <1>       ;
3854 00004558 0FB6C0                    <1>       movzx  eax, al
3855 0000455B 894504                    <1>       mov    [ebp+4], eax   ; ebx ; drive type (for floppy drives)
3856                                    <1>       ; 01/06/2016 (INT 33h, disk type return for floppy disks, in BL)
3857 0000455E A3[348D0100]              <1>       mov    [user_buffer], eax  ; 01/06/2016 (overwrite ebx return value)
3858                                    <1>       ;(INT 33h, Function 08h will replace user's buffer addr with disk type!)
3859                                    <1>       ;
3860 00004563 89DE                      <1>       mov    esi, ebx            ; floppy disk parameter table (16 bytes)
3861 00004565 B910000000                <1>       mov    ecx, 16 ; 16 bytes
3862 0000456A E8D3CF0000                <1>       call  transfer_to_user_buffer ; trdosk6.s (16/05/2016)
3863                                    <1> no_copy_fdpt:
3864 0000456F 5F                        <1>       pop    edi
3865                                    <1> DP_OUT:
3866 00004570 E85B020000                <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
3867 00004575 6631C0                    <1>       XOR    AX,AX               ; CLEAR
3868 00004578 F8                        <1>       CLC
3869 00004579 C3                        <1>       RETn
3870                                    <1>
3871                                    <1> ;-----      NO DRIVE PRESENT HANDLER
3872                                    <1>
3873                                    <1> NON_DRV:
3874                                    <1>       ;MOV   BYTE [BP+4],0       ; CLEAR NUMBER OF DRIVES
3875 0000457A 895508                    <1>       mov    [ebp+8], edx ; 0 ; 20/02/2015
3876                                    <1> NON_DRV1:
3877 0000457D 6681FF8000                <1>       CMP    DI,80H              ; CHECK FOR FIXED MEDIA TYPE REQUEST
3878 00004582 720C                      <1>       JB     short NON_DRV2      ; CONTINUE IF NOT REQUEST FALL THROUGH
3879                                    <1>
3880                                    <1> ;-----      FIXED DISK REQUEST FALL THROUGH ERROR
3881                                    <1>
3882 00004584 E847020000                <1>       CALL   XLAT_OLD            ; ELSE TRANSLATE TO COMPATIBLE MODE
3883 00004589 6689F0                    <1>       MOV    AX,SI               ; RESTORE AL
3884 0000458C B401                      <1>       MOV    AH,BAD_CMD          ; SET BAD COMMAND ERROR
3885 0000458E F9                        <1>       STC
3886 0000458F C3                        <1>       RETn
3887                                    <1>
3888                                    <1> NON_DRV2:
3889                                    <1>       ;XOR   AX,AX               ; CLEAR PARMS IF NO DRIVES OR CMOS BAD
3890 00004590 31C0                      <1>       xor    eax, eax
3891 00004592 66894500                  <1>       MOV    [eBP],AX            ; TRACKS, SECTORS/TRACK = 0
3892                                    <1>       ;MOV   [BP+5],AH           ; HEAD = 0
3893 00004596 886509                    <1>       mov    [ebp+9], ah ; 06/02/2015
3894                                    <1>       ;MOV   [BP+6],AX           ; OFFSET TO DISK_BASE = 0
3895 00004599 89450C                    <1>       mov    [ebp+12], eax
3896                                    <1>       ;;MOV  ES,AX               ; ES IS SEGMENT OF TABLE
3897                                    <1>       ;JMP   SHORT DP_OUT
3898                                    <1>
3899                                    <1>       ; 30/08/2020
3900 0000459C E82F020000                <1>       call   XLAT_OLD
3901                                    <1>       ;mov   ah, NOT_RDY ; drive not ready
3902 000045A1 B407                      <1>       mov    ah, INIT_FAIL ; DRIVE PARAMETER ACTIVITY FAILED
3903 000045A3 F9                        <1>       stc    ; cf -> 1, ah = 'drive not ready' error code
3904 000045A4 C3                        <1>       retn
3905                                    <1>
3906                                    <1> ;-----      DATA RATE IS EITHER 300 KBS OR 500 KBS, TRY 1.2 MB TABLE FIRST
3907                                    <1>
3908                                    <1> USE_EST2:
3909 000045A5 B002                      <1>       MOV    AL,02               ; DRIVE TYPE 2 (1.2MB)
3910 000045A7 E881010000                <1>       CALL   DR_TYPE_CHECK       ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
```

```
3911 000045AC 8A4B04              <1>       MOV   CL, [eBX+MD.SEC_TRK] ; GET SECTOR/TRACK
3912 000045AF 8A6B0B              <1>       MOV   CH, [eBX+MD.MAX_TRK] ; GET MAX. TRACK NUMBER
3913 000045B2 80FC40              <1>       CMP    AH,RATE_300         ; RATE 300 ?
3914 000045B5 7496                <1>       JZ     short STO_CX        ; MUST BE 1.2MB DRIVE
3915 000045B7 EB87                <1>       JMP    SHORT PARM144       ; ELSE, IT IS 1.44MB DRIVE
3916                              <1>
3917                              <1> ; 30/08/2020
3918                              <1>
3919                              <1> ;--------------------------------------------------------------------------
3920                              <1> ; DISK_TYPE (AH = 15H)
3921                              <1> ;     THIS ROUTINE RETURNS THE TYPE OF MEDIA INSTALLED.
3922                              <1> ;
3923                              <1> ;  ON ENTRY: DI = DRIVE #
3924                              <1> ;
3925                              <1> ;  ON EXIT:  AH = DRIVE TYPE, CY=0
3926                              <1> ;--------------------------------------------------------------------------
3927                              <1> DSK_TYPE:
3928 000045B9 E8E1010000          <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
3929 000045BE 8A87[3D810100]      <1>       MOV    AL,[DSK_STATE+eDI]  ; GET PRESENT STATE INFORMATION
3930 000045C4 08C0                <1>       OR     AL,AL               ; CHECK FOR NO DRIVE
3931 000045C6 7416                <1>       JZ     short NO_DRV
3932 000045C8 B401                <1>       MOV    AH,NOCHGLN          ; NO CHANGE LINE FOR 40 TRACK DRIVE
3933 000045CA A801                <1>       TEST   AL,TRK_CAPA         ; IS THIS DRIVE AN 80 TRACK DRIVE?
3934 000045CC 7402                <1>       JZ     short DT_BACK       ; IF NO JUMP
3935 000045CE B402                <1>       MOV    AH,CHGLN            ; CHANGE LINE FOR 80 TRACK DRIVE
3936                              <1> DT_BACK:
3937                              <1>       ;PUSH  AX                  ; SAVE RETURN VALUE
3938                              <1>       ; 11/04/2021
3939 000045D0 50                  <1>       push   eax
3940 000045D1 E8FA010000          <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
3941                              <1>       ; 11/04/2021
3942 000045D6 58                  <1>       pop    eax
3943 000045D7 F8                  <1>       CLC                        ; NO ERROR
3944 000045D8 6689F3              <1>       MOV    BX,SI               ; GET SAVED AL TO BL
3945 000045DB 88D8                <1>       MOV    AL,BL               ; PUT BACK FOR RETURN
3946 000045DD C3                  <1>       RETn
3947                              <1> NO_DRV:
3948                              <1>       ;XOR   AH,AH               ; NO DRIVE PRESENT OR UNKNOWN
3949                              <1>       ;JMP   SHORT DT_BACK
3950                              <1>
3951                              <1>       ; 30/08/2020
3952 000045DE E8ED010000          <1>       call   XLAT_OLD
3953 000045E3 29C0                <1>       sub    eax, eax
3954 000045E5 F9                  <1>       stc    ; cf = 1 -> drive not ready, ah = 0 (disk type = 0)
3955 000045E6 C3                  <1>       retn
3956                              <1>
3957                              <1> ;--------------------------------------------------------------------------
3958                              <1> ; DISK_CHANGE      (AH = 16H)
3959                              <1> ;     THIS ROUTINE RETURNS THE STATE OF THE DISK CHANGE LINE.
3960                              <1> ;
3961                              <1> ; ON ENTRY:  DI = DRIVE #
3962                              <1> ;
3963                              <1> ; ON EXIT:   AH = @DSKETTE_STATUS
3964                              <1> ;                 00 - DISK CHANGE LINE INACTIVE, CY = 0
3965                              <1> ;                 06 - DISK CHANGE LINE ACTIVE, CY = 1
3966                              <1> ;--------------------------------------------------------------------------
3967                              <1> DSK_CHANGE:
3968 000045E7 E8B3010000          <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
3969 000045EC 8A87[3D810100]      <1>       MOV    AL, [DSK_STATE+eDI] ; GET MEDIA STATE INFORMATION
3970 000045F2 08C0                <1>       OR     AL,AL               ; DRIVE PRESENT ?
3971 000045F4 7422                <1>       JZ     short DC_NON        ; JUMP IF NO DRIVE
3972 000045F6 A801                <1>       TEST   AL,TRK_CAPA         ; 80 TRACK DRIVE ?
3973 000045F8 7407                <1>       JZ     short SETIT         ; IF SO , CHECK CHANGE LINE
3974                              <1> DC0:
3975 000045FA E8640A0000          <1>       CALL   READ_DSKCHNG        ; GO CHECK STATE OF DISK CHANGE LINE
3976 000045FF 7407                <1>       JZ     short FINIS         ; CHANGE LINE NOT ACTIVE
3977                              <1>
3978 00004601 C605[30810100]06    <1> SETIT: MOV   byte [DSKETTE_STATUS], MEDIA_CHANGE ; INDICATE MEDIA REMOVED
3979                              <1>
3980 00004608 E8C3010000          <1> FINIS: CALL  XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
3981 0000460D E8E7060000          <1>       CALL   SETUP_END           ; VARIOUS CLEANUPS
3982 00004612 6689F3              <1>       MOV    BX,SI               ; GET SAVED AL TO BL
3983 00004615 88D8                <1>       MOV    AL,BL               ; PUT BACK FOR RETURN
3984 00004617 C3                  <1>       RETn
3985                              <1> DC_NON:
3986 00004618 800D[30810100]80    <1>       OR     byte [DSKETTE_STATUS], TIME_OUT ; SET TIMEOUT, NO DRIVE
3987 0000461F EBE7                <1>       JMP    SHORT FINIS
3988                              <1>
3989                              <1> ;--------------------------------------------------------------------------
3990                              <1> ; FORMAT_SET (AH = 17H)
3991                              <1> ;     THIS ROUTINE IS USED TO ESTABLISH THE TYPE OF MEDIA TO BE USED
3992                              <1> ;     FOR THE FOLLOWING FORMAT OPERATION.
3993                              <1> ;
3994                              <1> ; ON ENTRY:  SI LOW = DASD TYPE FOR FORMAT
3995                              <1> ;            DI     = DRIVE #
3996                              <1> ;
3997                              <1> ; ON EXIT:   @DSKETTE_STATUS REFLECTS STATUS
3998                              <1> ;            AH = @DSKETTE_STATUS
3999                              <1> ;            CY = 1 IF ERROR
4000                              <1> ;--------------------------------------------------------------------------
4001                              <1> FORMAT_SET:
4002 00004621 E879010000          <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
4003                              <1>       ;PUSH  SI                  ; SAVE DASD TYPE
4004                              <1>       ; 11/04/2021
4005 00004626 56                  <1>       push   esi
4006                              <1>       ;MOV   AX,SI               ; AH = ? , AL = DASD TYPE
4007                              <1>       ;XOR   AH,AH               ; AH = 0 , AL = DASD TYPE
4008                              <1>       ;MOV   SI,AX               ; SI = DASD TYPE
4009                              <1>       ; 11/04/2021
4010 00004627 89F0                <1>       mov    eax, esi
4011 00004629 0FB6F0              <1>       movzx  esi, al
4012 0000462C 80A7[3D810100]0F    <1>       AND    byte [DSK_STATE+eDI], ~(MED_DET+DBL_STEP+RATE_MSK) ; CLEAR STATE
4013                              <1>       ;DEC   SI                  ; CHECK FOR 320/360K MEDIA & DRIVE
4014                              <1>       ; 11/04/2021
4015 00004633 4E                  <1>       dec    esi
```

```
4016 00004634 7509               <1>       JNZ    short NOT_320       ; BYPASS IF NOT
4017 00004636 808F[3D810100]90   <1>       OR     byte [DSK_STATE+eDI], MED_DET+RATE_250 ; SET TO 320/360
4018 0000463D EB45               <1>       JMP    SHORT S0
4019                             <1>
4020                             <1> NOT_320:
4021 0000463F E8A4030000         <1>       CALL   MED_CHANGE          ; CHECK FOR TIME_OUT
4022 00004644 803D[30810100]80   <1>       CMP    byte [DSKETTE_STATUS], TIME_OUT
4023 0000464B 7437               <1>       JZ     short S0            ; IF TIME OUT TELL CALLER
4024                             <1> S3:
4025                             <1>       ;DEC   SI                  ; CHECK FOR 320/360K IN 1.2M DRIVE
4026                             <1>       ; 11/04/2021
4027 0000464D 4E                 <1>       dec    esi
4028 0000464E 7509               <1>       JNZ    short NOT_320_12    ; BYPASS IF NOT
4029 00004650 808F[3D810100]70   <1>       OR     byte [DSK_STATE+eDI], MED_DET+DBL_STEP+RATE_300 ; SET STATE
4030 00004657 EB2B               <1>       JMP    SHORT S0
4031                             <1>
4032                             <1> NOT_320_12:
4033                             <1>       ;DEC   SI                  ; CHECK FOR 1.2M MEDIA IN 1.2M DRIVE
4034                             <1>       ; 11/04/2021
4035 00004659 4E                 <1>       dec    esi
4036 0000465A 7509               <1>       JNZ    short NOT_12        ; BYPASS IF NOT
4037 0000465C 808F[3D810100]10   <1>       OR     byte [DSK_STATE+eDI], MED_DET+RATE_500 ; SET STATE VARIABLE
4038 00004663 EB1F               <1>       JMP    SHORT S0            ; RETURN TO CALLER
4039                             <1>
4040                             <1> NOT_12:
4041                             <1>       ;DEC   SI                  ; CHECK FOR SET DASD TYPE 04
4042                             <1>       ; 11/04/2021
4043 00004665 4E                 <1>       dec    esi
4044 00004666 752A               <1>       JNZ    short FS_ERR        ; BAD COMMAND EXIT IF NOT VALID TYPE
4045                             <1>
4046 00004668 F687[3D810100]04   <1>       TEST   byte [DSK_STATE+eDI], DRV_DET ; DRIVE DETERMINED ?
4047 0000466F 740B               <1>       JZ     short ASSUME        ; IF STILL NOT DETERMINED ASSUME
4048 00004671 B050               <1>       MOV    AL,MED_DET+RATE_300
4049 00004673 F687[3D810100]02   <1>       TEST  byte [DSK_STATE+eDI], FMT_CAPA ; MULTIPLE FORMAT CAPABILITY ?
4050 0000467A 7502               <1>       JNZ    short OR_IT_IN            ; IF 1.2 M THEN DATA RATE 300
4051                             <1>
4052                             <1> ASSUME:
4053 0000467C B090               <1>       MOV    AL,MED_DET+RATE_250 ; SET UP
4054                             <1>
4055                             <1> OR_IT_IN:
4056 0000467E 0887[3D810100]     <1>       OR     [DSK_STATE+eDI], AL ; OR IN THE CORRECT STATE
4057                             <1> S0:
4058 00004684 E847010000         <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
4059 00004689 E86B060000         <1>       CALL   SETUP_END           ; VARIOUS CLEANUPS
4060                             <1>       ;POP   BX                  ; GET SAVED AL TO BL
4061                             <1>       ; 11/04/2021
4062 0000468E 5B                 <1>       pop    ebx
4063 0000468F 88D8               <1>       MOV    AL,BL               ; PUT BACK FOR RETURN
4064 00004691 C3                 <1>       RETn
4065                             <1>
4066                             <1> FS_ERR:
4067 00004692 C605[30810100]01   <1>       MOV    byte [DSKETTE_STATUS], BAD_CMD ; UNKNOWN STATE,BAD COMMAND
4068 00004699 EBE9               <1>       JMP    SHORT S0
4069                             <1>
4070                             <1> ;--------------------------------------------------------------------------------
4071                             <1> ; SET_MEDIA  (AH = 18H)
4072                             <1> ;     THIS ROUTINE SETS THE TYPE OF MEDIA AND DATA RATE
4073                             <1> ;     TO BE USED FOR THE FOLLOWING FORMAT OPERATION.
4074                             <1> ;
4075                             <1> ; ON ENTRY:
4076                             <1> ;     [BP]   = SECTOR PER TRACK
4077                             <1> ;     [BP+1] = TRACK #
4078                             <1> ;     DI     = DRIVE #
4079                             <1> ;
4080                             <1> ; ON EXIT:
4081                             <1> ;     @DSKETTE_STATUS REFLECTS STATUS
4082                             <1> ;     IF NO ERROR:
4083                             <1> ;           AH = 0
4084                             <1> ;           CY = 0
4085                             <1> ;           ES = SEGMENT OF MEDIA/DRIVE PARAMETER TABLE
4086                             <1> ;           DI/[BP+6] = OFFSET OF MEDIA/DRIVE PARAMETER TABLE
4087                             <1> ;     IF ERROR:
4088                             <1> ;           AH = @DSKETTE_STATUS
4089                             <1> ;           CY = 1
4090                             <1> ;--------------------------------------------------------------------------------
4091                             <1> SET_MEDIA:
4092 0000469B E8FF000000         <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
4093 000046A0 F687[3D810100]01   <1>         TEST    byte [DSK_STATE+eDI], TRK_CAPA ; CHECK FOR CHANGE LINE AVAILABLE
4094 000046A7 7415               <1>       JZ     short SM_CMOS       ; JUMP IF 40 TRACK DRIVE
4095 000046A9 E83A030000         <1>       CALL   MED_CHANGE          ; RESET CHANGE LINE
4096 000046AE 803D[30810100]80   <1>       CMP    byte [DSKETTE_STATUS], TIME_OUT ; IF TIME OUT TELL CALLER
4097 000046B5 746B               <1>       JE     short SM_RTN
4098 000046B7 C605[30810100]00   <1>       MOV    byte [DSKETTE_STATUS], 0 ; CLEAR STATUS
4099                             <1> SM_CMOS:
4100 000046BE E8FB060000         <1>       CALL   CMOS_TYPE           ; RETURN DRIVE TYPE IN (AL)
4101                             <1>       ;;20/02/2015
4102                             <1>       ;;JC   short MD_NOT_FND    ; ERROR IN CMOS
4103                             <1>       ;;OR   AL,AL               ; TEST FOR NO DRIVE
4104 000046C3 745D               <1>       JZ     short SM_RTN        ; RETURN IF SO
4105 000046C5 E863000000         <1>       CALL   DR_TYPE_CHECK       ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
4106 000046CA 7231               <1>       JC     short MD_NOT_FND    ; TYPE NOT IN TABLE (BAD CMOS)
4107 000046CC 57                 <1>       PUSH   eDI                 ; SAVE REG.
4108 000046CD 31DB               <1>       XOR    eBX,eBX             ; BX = INDEX TO DR. TYPE TABLE
4109 000046CF B906000000         <1>       MOV    eCX,DR_CNT          ; CX = LOOP COUNT
4110                             <1> DR_SEARCH:
4111 000046D4 8AA3[68680000]     <1>       MOV    AH, [DR_TYPE+eBX]   ; GET DRIVE TYPE
4112 000046DA 80E47F             <1>       AND    AH,BIT7OFF          ; MASK OUT MSB
4113 000046DD 38E0               <1>       CMP    AL,AH               ; DRIVE TYPE MATCH ?
4114 000046DF 7516               <1>       JNE    short NXT_MD        ; NO, CHECK NEXT DRIVE TYPE
4115                             <1> DR_FND:
4116 000046E1 8BBB[69680000]     <1>       MOV    eDI, [DR_TYPE+eBX+1] ; DI = MEDIA/DRIVE PARAM TABLE
4117                             <1> MD_SEARCH:
4118 000046E7 8A6704             <1>       MOV   AH, [eDI+MD.SEC_TRK] ; GET SECTOR/TRACK
4119 000046EA 386500             <1>       CMP    [eBP],AH            ; MATCH?
4120 000046ED 7508               <1>       JNE    short NXT_MD        ; NO, CHECK NEXT MEDIA
```

```
4121 000046EF 8A670B              <1>       MOV    AH, [eDI+MD.MAX_TRK] ; GET MAX. TRACK #
4122 000046F2 386501              <1>       CMP    [eBP+1],AH           ; MATCH?
4123 000046F5 740F                <1>       JE     short MD_FND         ; YES, GO GET RATE
4124                              <1> NXT_MD:
4125                              <1>       ;ADD   BX,3                 ; CHECK NEXT DRIVE TYPE
4126 000046F7 83C305              <1>       add    ebx, 5 ; 18/02/2015
4127 000046FA E2D8                <1>       LOOP  DR_SEARCH
4128 000046FC 5F                  <1>       POP    eDI                  ; RESTORE REG.
4129                              <1> MD_NOT_FND:
4130 000046FD C605[30810100]0C    <1>       MOV    byte [DSKETTE_STATUS], MED_NOT_FND ; ERROR, MEDIA TYPE NOT FOUND
4131 00004704 EB1C                <1>       JMP    SHORT SM_RTN         ; RETURN
4132                              <1> MD_FND:
4133 00004706 8A470C              <1>       MOV   AL, [eDI+MD.RATE]    ; GET RATE
4134 00004709 3C40                <1>       CMP    AL,RATE_300          ; DOUBLE STEP REQUIRED FOR RATE 300
4135 0000470B 7502                <1>       JNE    short MD_SET
4136 0000470D 0C20                <1>       OR     AL,DBL_STEP
4137                              <1> MD_SET:
4138                              <1>       ;MOV   [BP+6],DI           ; SAVE TABLE POINTER IN STACK
4139 0000470F 897D0C              <1>       mov    [ebp+12], edi ; 18/02/2015
4140 00004712 0C10                <1>       OR     AL,MED_DET           ; SET MEDIA ESTABLISHED
4141 00004714 5F                  <1>       POP    eDI
4142 00004715 80A7[3D810100]0F    <1>       AND    byte [DSK_STATE+eDI], ~(MED_DET+DBL_STEP+RATE_MSK) ; CLEAR STATE
4143 0000471C 0887[3D810100]      <1>       OR     [DSK_STATE+eDI], AL
4144                              <1>       ;MOV   AX, CS              ; SEGMENT OF MEDIA/DRIVE PARAMETER TABLE
4145                              <1>       ;MOV   ES, AX              ; ES IS SEGMENT OF TABLE
4146                              <1> SM_RTN:
4147 00004722 E8A9000000          <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
4148 00004727 E8CD050000          <1>       CALL   SETUP_END           ; VARIOUS CLEANUPS
4149 0000472C C3                  <1>       RETn
4150                              <1>
4151                              <1> ;--------------------------------------------------------------------
4152                              <1> ; DR_TYPE_CHECK                                                    :
4153                              <1> ;     CHECK IF THE GIVEN DRIVE TYPE IN REGISTER (AL)               :
4154                              <1> ;     IS SUPPORTED IN BIOS DRIVE TYPE TABLE                        :
4155                              <1> ; ON ENTRY:                                                        :
4156                              <1> ;     AL = DRIVE TYPE                                              :
4157                              <1> ; ON EXIT:                                                         :
4158                              <1> ;     CS = SEGMENT MEDIA/DRIVE PARAMETER TABLE (CODE)              :
4159                              <1> ;     CY = 0       DRIVE TYPE SUPPORTED                            :
4160                              <1> ;          BX = OFFSET TO MEDIA/DRIVE PARAMETER TABLE              :
4161                              <1> ;     CY = 1 DRIVE TYPE NOT SUPPORTED                              :
4162                              <1> ; REGISTERS ALTERED: eBX                                           :
4163                              <1> ;--------------------------------------------------------------------
4164                              <1> DR_TYPE_CHECK:
4165                              <1>       ;PUSH  AX
4166                              <1>       ; 11/04/2021
4167 0000472D 50                  <1>       push   eax
4168 0000472E 51                  <1>       PUSH   eCX
4169 0000472F 31DB                <1>       XOR    eBX,eBX             ; BX = INDEX TO DR_TYPE TABLE
4170 00004731 B906000000          <1>       MOV    eCX,DR_CNT          ; CX = LOOP COUNT
4171                              <1> TYPE_CHK:
4172 00004736 8AA3[68680000]      <1>       MOV    AH,[DR_TYPE+eBX]    ; GET DRIVE TYPE
4173 0000473C 38E0                <1>       CMP    AL,AH               ; DRIVE TYPE MATCH?
4174 0000473E 740D                <1>       JE     short DR_TYPE_VALID ; YES, RETURN WITH CARRY RESET
4175                              <1>       ;ADD   BX,3                ; CHECK NEXT DRIVE TYPE
4176 00004740 83C305              <1>       add    ebx, 5 ; 16/02/2015 (32 bit address modification)
4177 00004743 E2F1                <1>       LOOP  TYPE_CHK
4178                              <1>       ;
4179 00004745 BB[C7680000]        <1>       mov    ebx, MD_TBL6        ; 1.44MB fd parameter table
4180                              <1>                                  ; Default for GET_PARM (11/12/2014)
4181                              <1>       ;
4182 0000474A F9                  <1>       STC                        ; DRIVE TYPE NOT FOUND IN TABLE
4183 0000474B EB06                <1>       JMP    SHORT TYPE_RTN
4184                              <1> DR_TYPE_VALID:
4185 0000474D 8B9B[69680000]      <1>       MOV    eBX,[DR_TYPE+eBX+1] ; BX = MEDIA TABLE
4186                              <1> TYPE_RTN:
4187 00004753 59                  <1>       POP    eCX
4188                              <1>       ;POP   AX
4189                              <1>       ; 11/04/2021
4190 00004754 58                  <1>       pop    eax
4191 00004755 C3                  <1>       RETn
4192                              <1>
4193                              <1> ;--------------------------------------------------------------------
4194                              <1> ; SEND_SPEC                                                        :
4195                              <1> ;     SEND THE SPECIFY COMMAND TO CONTROLLER USING DATA FROM        :
4196                              <1> ;     THE DRIVE PARAMETER TABLE POINTED BY @DISK_POINTER            :
4197                              <1> ; ON ENTRY:  @DISK_POINTER = DRIVE PARAMETER TABLE                 :
4198                              <1> ; ON EXIT:   NONE                                                  :
4199                              <1> ; REGISTERS ALTERED: CX, DX                                        :
4200                              <1> ;--------------------------------------------------------------------
4201                              <1> SEND_SPEC:
4202 00004756 50                  <1>       PUSH   eAX                 ; SAVE AX
4203 00004757 B8[7D470000]        <1>       MOV    eAX, SPECBAC        ; LOAD ERROR ADDRESS
4204 0000475C 50                  <1>       PUSH   eAX                 ; PUSH NEC_OUT ERROR RETURN
4205 0000475D B403                <1>       MOV    AH,03H              ; SPECIFY COMMAND
4206 0000475F E869070000          <1>       CALL   NEC_OUTPUT          ; OUTPUT THE COMMAND
4207 00004764 28D2                <1>       SUB    DL,DL               ; FIRST SPECIFY BYTE
4208 00004766 E85C060000          <1>       CALL   GET_PARM            ; GET PARAMETER TO AH
4209 0000476B E85D070000          <1>       CALL   NEC_OUTPUT          ; OUTPUT THE COMMAND
4210 00004770 B201                <1>       MOV    DL,1                ; SECOND SPECIFY BYTE
4211 00004772 E850060000          <1>       CALL   GET_PARM            ; GET PARAMETER TO AH
4212 00004777 E851070000          <1>       CALL   NEC_OUTPUT          ; OUTPUT THE COMMAND
4213 0000477C 58                  <1>       POP    eAX                 ; POP ERROR RETURN
4214                              <1> SPECBAC:
4215 0000477D 58                  <1>       POP    eAX                 ; RESTORE ORIGINAL AX VALUE
4216 0000477E C3                  <1>       RETn
4217                              <1>
4218                              <1> ;--------------------------------------------------------------------
4219                              <1> ; SEND_SPEC_MD                                                     :
4220                              <1> ;     SEND THE SPECIFY COMMAND TO CONTROLLER USING DATA FROM        :
4221                              <1> ;     THE MEDIA/DRIVE PARAMETER TABLE POINTED BY (CS:BX)           :
4222                              <1> ; ON ENTRY:  CS:BX = MEDIA/DRIVE PARAMETER TABLE                   :
4223                              <1> ; ON EXIT:   NONE                                                  :
4224                              <1> ; REGISTERS ALTERED: AX                                            :
4225                              <1> ;--------------------------------------------------------------------
```

```
                        <1> SEND_SPEC_MD:
0000477F 50             <1>         PUSH   eAX                  ; SAVE RATE DATA
00004780 B8[9D470000]   <1>         MOV    eAX, SPEC_ESBAC      ; LOAD ERROR ADDRESS
00004785 50             <1>         PUSH   eAX                  ; PUSH NEC_OUT ERROR RETURN
00004786 B403           <1>         MOV    AH,03H               ; SPECIFY COMMAND
00004788 E840070000     <1>         CALL   NEC_OUTPUT           ; OUTPUT THE COMMAND
0000478D 8A23           <1>         MOV   AH, [eBX+MD.SPEC1]    ; GET 1ST SPECIFY BYTE
0000478F E839070000     <1>         CALL   NEC_OUTPUT           ; OUTPUT THE COMMAND
00004794 8A6301         <1>         MOV   AH, [eBX+MD.SPEC2]    ; GET SECOND SPECIFY BYTE
00004797 E831070000     <1>         CALL   NEC_OUTPUT           ; OUTPUT THE COMMAND
0000479C 58             <1>         POP    eAX                  ; POP ERROR RETURN
                        <1> SPEC_ESBAC:
0000479D 58             <1>         POP    eAX                  ; RESTORE ORIGINAL AX VALUE
0000479E C3             <1>         RETn
                        <1>
                        <1> ;--------------------------------------------------------------------------
                        <1> ; XLAT_NEW
                        <1> ;     TRANSLATES DISKETTE STATE LOCATIONS FROM COMPATIBLE
                        <1> ;     MODE TO NEW ARCHITECTURE.
                        <1> ;
                        <1> ; ON ENTRY:  DI = DRIVE #
                        <1> ;--------------------------------------------------------------------------
                        <1> XLAT_NEW:
0000479F 83FF01         <1>         CMP    eDI,1                    ; VALID DRIVE
000047A2 7725           <1>         JA     short XN_OUT             ; IF INVALID BACK
000047A4 80BF[3D810100]00 <1>       CMP    byte [DSK_STATE+eDI], 0  ; NO DRIVE ?
000047AB 741D           <1>         JZ     short DO_DET             ; IF NO DRIVE ATTEMPT DETERMINE
000047AD 6689F9         <1>         MOV    CX,DI                    ; CX = DRIVE NUMBER
000047B0 C0E102         <1>         SHL    CL,2                     ; CL = SHIFT COUNT, A=0, B=4
000047B3 A0[3C810100]   <1>         MOV    AL, [HF_CNTRL]           ; DRIVE INFORMATION
000047B8 D2C8           <1>         ROR    AL,CL                    ; TO LOW NIBBLE
000047BA 2407           <1>         AND    AL,DRV_DET+FMT_CAPA+TRK_CAPA ; KEEP DRIVE BITS
000047BC 80A7[3D810100]F8 <1>       AND   byte [DSK_STATE+eDI], ~(DRV_DET+FMT_CAPA+TRK_CAPA)
000047C3 0887[3D810100] <1>         OR     [DSK_STATE+eDI], AL       ; UPDATE DRIVE STATE
                        <1> XN_OUT:
000047C9 C3             <1>         RETn
                        <1> DO_DET:
000047CA E8A1080000     <1>         CALL   DRIVE_DET                ; TRY TO DETERMINE
000047CF C3             <1>         RETn
                        <1>
                        <1> ;--------------------------------------------------------------------------
                        <1> ; XLAT_OLD
                        <1> ;     TRANSLATES DISKETTE STATE LOCATIONS FROM NEW
                        <1> ;     ARCHITECTURE TO COMPATIBLE MODE.
                        <1> ;
                        <1> ; ON ENTRY:  DI = DRIVE
                        <1> ;--------------------------------------------------------------------------
                        <1> XLAT_OLD:
000047D0 83FF01         <1>         CMP    eDI,1                ; VALID DRIVE ?
                        <1>         ;JA   short XO_OUT          ; IF INVALID BACK
000047D3 0F8786000000   <1>         ja    XO_OUT
000047D9 80BF[3D810100]00 <1>       CMP    byte [DSK_STATE+eDI],0 ; NO DRIVE ?
000047E0 747D           <1>         JZ     short XO_OUT         ; IF NO DRIVE TRANSLATE DONE
                        <1>
                        <1> ;-----      TEST FOR SAVED DRIVE INFORMATION ALREADY SET
                        <1>
000047E2 6689F9         <1>         MOV    CX,DI                ; CX = DRIVE NUMBER
000047E5 C0E102         <1>         SHL    CL,2                 ; CL = SHIFT COUNT, A=0, B=4
000047E8 B402           <1>         MOV    AH,FMT_CAPA          ; LOAD MULTIPLE DATA RATE BIT MASK
000047EA D2CC           <1>         ROR    AH,CL                ; ROTATE BY MASK
000047EC 8425[3C810100] <1>         TEST   [HF_CNTRL], AH       ; MULTIPLE-DATA RATE DETERMINED ?
000047F2 751C           <1>         JNZ    short SAVE_SET       ; IF SO, NO NEED TO RE-SAVE
                        <1>
                        <1> ;-----      ERASE DRIVE BITS IN @HF_CNTRL FOR THIS DRIVE
                        <1>
000047F4 B407           <1>         MOV    AH,DRV_DET+FMT_CAPA+TRK_CAPA ; MASK TO KEEP
000047F6 D2CC           <1>         ROR    AH,CL                ; FIX MASK TO KEEP
000047F8 F6D4           <1>         NOT    AH                   ; TRANSLATE MASK
000047FA 2025[3C810100] <1>         AND    [HF_CNTRL], AH       ; KEEP BITS FROM OTHER DRIVE INTACT
                        <1>
                        <1> ;-----      ACCESS CURRENT DRIVE BITS AND STORE IN @HF_CNTRL
                        <1>
00004800 8A87[3D810100] <1>         MOV    AL, [DSK_STATE+eDI] ; ACCESS STATE
00004806 2407           <1>         AND    AL,DRV_DET+FMT_CAPA+TRK_CAPA ; KEEP DRIVE BITS
00004808 D2C8           <1>         ROR    AL,CL                ; FIX FOR THIS DRIVE
0000480A 0805[3C810100] <1>         OR     [HF_CNTRL], AL       ; UPDATE SAVED DRIVE STATE
                        <1>
                        <1> ;-----      TRANSLATE TO COMPATIBILITY MODE
                        <1>
                        <1> SAVE_SET:
00004810 8AA7[3D810100] <1>         MOV    AH, [DSK_STATE+eDI] ; ACCESS STATE
00004816 88E7           <1>         MOV    BH,AH                ; TO BH FOR LATER
00004818 80E4C0         <1>         AND    AH,RATE_MSK          ; KEEP ONLY RATE
0000481B 80FC00         <1>         CMP    AH,RATE_500          ; RATE 500 ?
0000481E 7410           <1>         JZ     short CHK_144        ; YES 1.2/1.2 OR 1.44/1.44
00004820 B001           <1>         MOV    AL,M3D1U             ; AL = 360 IN 1.2 UNESTABLISHED
00004822 80FC40         <1>         CMP    AH,RATE_300          ; RATE 300 ?
00004825 7518           <1>         JNZ    short CHK_250        ; NO, 360/360, 720/720 OR 720/1.44
00004827 F6C720         <1>         TEST   BH,DBL_STEP          ; CHECK FOR DOUBLE STEP
0000482A 751F           <1>         JNZ    short TST_DET        ; MUST BE 360 IN 1.2
                        <1> UNKNO:
0000482C B007           <1>         MOV    AL,MED_UNK           ; NONE OF THE ABOVE
0000482E EB22           <1>         JMP    SHORT AL_SET         ; PROCESS COMPLETE
                        <1> CHK_144:
00004830 E889050000     <1>         CALL   CMOS_TYPE            ; RETURN DRIVE TYPE IN (AL)
                        <1>         ;;20/02/2015
                        <1>         ;;JC   short UNKNO          ; ERROR, SET 'NONE OF ABOVE'
00004835 74F5           <1>         jz     short UNKNO ;; 20/02/2015
00004837 3C02           <1>         CMP    AL,2                 ; 1.2MB DRIVE ?
00004839 75F1           <1>         JNE    short UNKNO          ; NO, GO SET 'NONE OF ABOVE'
0000483B B002           <1>         MOV    AL,M1D1U             ; AL = 1.2 IN 1.2 UNESTABLISHED
0000483D EB0C           <1>         JMP    SHORT TST_DET
                        <1> CHK_250:
0000483F B000           <1>         MOV    AL,M3D3U             ; AL = 360 IN 360 UNESTABLISHED
00004841 80FC80         <1>         CMP    AH,RATE_250          ; RATE 250 ?
```

```
4331 00004844 75E6               <1>       JNZ    short UNKNO         ; IF SO FALL IHRU
4332 00004846 F6C701             <1>       TEST   BH,TRK_CAPA         ; 80 TRACK CAPABILITY ?
4333 00004849 75E1               <1>       JNZ    short UNKNO         ; IF SO JUMP, FALL THRU TEST DET
4334                             <1> TST_DET:
4335 0000484B F6C710             <1>       TEST   BH,MED_DET          ; DETERMINED ?
4336 0000484E 7402               <1>       JZ     short AL_SET        ; IF NOT THEN SET
4337 00004850 0403               <1>       ADD    AL,3                ; MAKE DETERMINED/ESTABLISHED
4338                             <1> AL_SET:
4339 00004852 80A7[3D810100]F8   <1>       AND    byte [DSK_STATE+eDI], ~(DRV_DET+FMT_CAPA+TRK_CAPA) ; CLEAR DRIVE
4340 00004859 0887[3D810100]     <1>       OR     [DSK_STATE+eDI], AL ; REPLACE WITH COMPATIBLE MODE
4341                             <1> XO_OUT:
4342 0000485F C3                 <1>       RETn
4343                             <1>
4344                             <1> ;--------------------------------------------------------------------------
4345                             <1> ; RD_WR_VF
4346                             <1> ;     COMMON READ, WRITE AND VERIFY:
4347                             <1> ;     MAIN LOOP FOR STATE RETRIES.
4348                             <1> ;
4349                             <1> ; ON ENTRY:  AH = READ/WRITE/VERIFY NEC PARAMETER
4350                             <1> ;            AL = READ/WRITE/VERIFY DMA PARAMETER
4351                             <1> ;
4352                             <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
4353                             <1> ;--------------------------------------------------------------------------
4354                             <1> RD_WR_VF:
4355                             <1>       ; 11/04/2021 (32 bit push/pop, AX -> eAX)
4356 00004860 50                 <1>       PUSH   eAX                 ; SAVE DMA, NEC PARAMETERS
4357 00004861 E839FFFFFF         <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
4358 00004866 E8E8000000         <1>       CALL   SETUP_STATE         ; INITIALIZE START AND END RATE
4359 0000486B 58                 <1>       POP    eAX                 ; RESTORE READ/WRITE/VERIFY
4360                             <1> DO_AGAIN:
4361 0000486C 50                 <1>       PUSH   eAX                 ; SAVE READ/WRITE/VERIFY PARAMETER
4362 0000486D E876010000         <1>       CALL   MED_CHANGE          ; MEDIA CHANGE AND RESET IF CHANGED
4363 00004872 58                 <1>       POP    eAX                 ; RESTORE READ/WRITE/VERIFY
4364 00004873 0F82C3000000       <1>       JC     RWV_END             ; MEDIA CHANGE ERROR OR TIME-OUT
4365                             <1> RWV:
4366 00004879 50                 <1>       PUSH   eAX                 ; SAVE READ/WRITE/VERIFY PARAMETER
4367 0000487A 8AB7[3D810100]     <1>       MOV    DH, [DSK_STATE+eDI] ; GET RATE STATE OF THIS DRIVE
4368 00004880 80E6C0             <1>       AND    DH,RATE_MSK         ; KEEP ONLY RATE
4369 00004883 E836050000         <1>       CALL   CMOS_TYPE           ; RETURN DRIVE TYPE IN AL (AL)
4370                             <1>       ;;20/02/2015
4371                             <1>       ;;JC   short RWV_ASSUME    ; ERROR IN CMOS
4372 00004888 7451               <1>       jz     short RWV_ASSUME ; 20/02/2015
4373 0000488A 3C01               <1>       CMP    AL,1                ; 40 TRACK DRIVE?
4374 0000488C 750D               <1>       JNE    short RWV_1         ; NO, BYPASS CMOS VALIDITY CHECK
4375 0000488E F687[3D810100]01   <1>       TEST   byte [DSK_STATE+eDI], TRK_CAPA ; CHECK FOR 40 TRACK DRIVE
4376 00004895 7413               <1>       JZ     short RWV_2         ; YES, CMOS IS CORRECT
4377 00004897 B002               <1>       MOV    AL,2                ; CHANGE TO 1.2M
4378 00004899 EB0F               <1>       JMP    SHORT RWV_2
4379                             <1> RWV_1:
4380 0000489B 720D               <1>       JB     short RWV_2         ; NO DRIVE SPECIFIED, CONTINUE
4381 0000489D F687[3D810100]01   <1>       TEST    byte [DSK_STATE+eDI], TRK_CAPA ; IS IT REALLY 40 TRACK?
4382 000048A4 7504               <1>       JNZ    short RWV_2         ; NO, 80 TRACK
4383 000048A6 B001               <1>       MOV    AL,1                ; IT IS 40 TRACK, FIX CMOS VALUE
4384 000048A8 EB04               <1>       jmp    short rwv_3
4385                             <1> RWV_2:
4386 000048AA 08C0               <1>       OR     AL,AL               ; TEST FOR NO DRIVE
4387 000048AC 742D               <1>       JZ     short RWV_ASSUME    ; ASSUME TYPE, USE MAX TRACK
4388                             <1> rwv_3:
4389 000048AE E87AFEFFFF         <1>       CALL   DR_TYPE_CHECK       ; RTN CS:BX = MEDIA/DRIVE PARAM TBL.
4390 000048B3 7226               <1>       JC     short RWV_ASSUME    ; TYPE NOT IN TABLE (BAD CMOS)
4391                             <1>
4392                             <1> ;----- SEARCH FOR MEDIA/DRIVE PARAMETER TABLE
4393                             <1>
4394 000048B5 57                 <1>       PUSH   eDI                 ; SAVE DRIVE #
4395 000048B6 31DB               <1>       XOR    eBX,eBX                   ; BX = INDEX TO DR_TYPE TABLE
4396 000048B8 B906000000         <1>       MOV    eCX,DR_CNT          ; CX = LOOP COUNT
4397                             <1> RWV_DR_SEARCH:
4398 000048BD 8AA3[68680000]     <1>       MOV    AH, [DR_TYPE+eBX]   ; GET DRIVE TYPE
4399 000048C3 80E47F             <1>       AND    AH,BIT7OFF          ; MASK OUT MSB
4400 000048C6 38E0               <1>       CMP    AL,AH               ; DRIVE TYPE MATCH?
4401 000048C8 750B               <1>       JNE    short RWV_NXT_MD    ; NO, CHECK NEXT DRIVE TYPE
4402                             <1> RWV_DR_FND:
4403 000048CA 8BBB[69680000]     <1>       MOV    eDI, [DR_TYPE+eBX+1] ; DI = MEDIA/DRIVE PARAMETER TABLE
4404                             <1> RWV_MD_SEARH:
4405 000048D0 3A770C             <1>       CMP   DH, [eDI+MD.RATE]    ; MATCH?
4406 000048D3 741B               <1>       JE     short RWV_MD_FND    ; YES, GO GET 1ST SPECIFY BYTE
4407                             <1> RWV_NXT_MD:
4408                             <1>       ;ADD   BX,3                ; CHECK NEXT DRIVE TYPE
4409 000048D5 83C305             <1>       add    eBX, 5
4410 000048D8 E2E3               <1>       LOOP   RWV_DR_SEARCH
4411 000048DA 5F                 <1>       POP    eDI                 ; RESTORE DRIVE #
4412                             <1>
4413                             <1> ;----- ASSUME PRIMARY DRIVE IS INSTALLED AS SHIPPED
4414                             <1>
4415                             <1> RWV_ASSUME:
4416 000048DB BB[86680000]       <1>       MOV    eBX, MD_TBL1        ; POINT TO 40 TRACK 250 KBS
4417 000048E0 F687[3D810100]01   <1>       TEST   byte [DSK_STATE+eDI], TRK_CAPA ; TEST FOR 80 TRACK
4418 000048E7 740A               <1>       JZ     short RWV_MD_FND1   ; MUST BE 40 TRACK
4419 000048E9 BB[A0680000]       <1>       MOV    eBX, MD_TBL3        ; POINT TO 80 TRACK 500 KBS
4420 000048EE EB03               <1>       JMP    short RWV_MD_FND1   ; GO SPECIFY PARAMTERS
4421                             <1>
4422                             <1> ;----- CS:BX POINTS TO MEDIA/DRIVE PARAMETER TABLE
4423                             <1>
4424                             <1> RWV_MD_FND:
4425 000048F0 89FB               <1>       MOV    eBX,eDI             ; BX = MEDIA/DRIVE PARAMETER TABLE
4426 000048F2 5F                 <1>       POP    eDI                 ; RESTORE DRIVE #
4427                             <1>
4428                             <1> ;----- SEND THE SPECIFY COMMAND TO THE CONTROLLER
4429                             <1>
4430                             <1> RWV_MD_FND1:
4431 000048F3 E887FEFFFF         <1>       CALL   SEND_SPEC_MD
4432 000048F8 E85B010000         <1>       CALL   CHK_LASTRATE        ; ZF=1 ATTEMP RATE IS SAME AS LAST RATE
4433 000048FD 7405               <1>       JZ     short RWV_DBL       ; YES,SKIP SEND RATE COMMAND
4434 000048FF E834010000         <1>       CALL   SEND_RATE           ; SEND DATA RATE TO NEC
4435                             <1> RWV_DBL:
```

```
4436 00004904 53                <1>       PUSH   eBX                ; SAVE MEDIA/DRIVE PARAM TBL ADDRESS
4437 00004905 E80F040000        <1>       CALL   SETUP_DBL          ; CHECK FOR DOUBLE STEP
4438 0000490A 5B                <1>       POP    eBX                ; RESTORE ADDRESS
4439 0000490B 7222              <1>       JC     short CHK_RET      ; ERROR FROM READ ID, POSSIBLE RETRY
4440 0000490D 58                <1>       POP    eAX                ; RESTORE NEC, DMA COMMAND
4441 0000490E 50                <1>       PUSH   eAX                ; SAVE NEC COMMAND
4442 0000490F 53                <1>       PUSH   eBX                ; SAVE MEDIA/DRIVE PARAM TBL ADDRESS
4443 00004910 E858010000        <1>       CALL   DMA_SETUP          ; SET UP THE DMA
4444 00004915 5B                <1>       POP    eBX
4445 00004916 58                <1>       POP    eAX                ; RESTORE NEC COMMAND
4446 00004917 722D              <1>       JC     short RWV_BAC      ; CHECK FOR DMA BOUNDARY ERROR
4447 00004919 50                <1>       PUSH   eAX                ; SAVE NEC COMMAND
4448 0000491A 53                <1>       PUSH   eBX                ; SAVE MEDIA/DRIVE PARAM TBL ADDRESS
4449 0000491B E831020000        <1>       CALL   NEC_INIT           ; INITIALIZE NEC
4450 00004920 5B                <1>       POP    eBX                ; RESTORE ADDRESS
4451 00004921 720C              <1>       JC     short CHK_RET      ; ERROR - EXIT
4452 00004923 E859020000        <1>       CALL   RWV_COM            ; OP CODE COMMON TO READ/WRITE/VERIFY
4453 00004928 7205              <1>       JC     short CHK_RET      ; ERROR - EXIT
4454 0000492A E89E020000        <1>       CALL   NEC_TERM           ; TERMINATE, GET STATUS, ETC.
4455                            <1> CHK_RET:
4456 0000492F E83D030000        <1>       CALL   RETRY              ; CHECK FOR, SETUP RETRY
4457 00004934 58                <1>       POP    eAX                ; RESTORE READ/WRITE/VERIFY PARAMETER
4458 00004935 7305              <1>       JNC    short RWV_END      ; CY = 0 NO RETRY
4459 00004937 E930FFFFFF        <1>       JMP   DO_AGAIN            ; CY = 1 MEANS RETRY
4460                            <1> RWV_END:
4461 0000493C E8E8020000        <1>       CALL   DSTATE             ; ESTABLISH STATE IF SUCCESSFUL
4462 00004941 E87B030000        <1>       CALL   NUM_TRANS          ; AL = NUMBER TRANSFERRED
4463                            <1> RWV_BAC:                        ; BAD DMA ERROR ENTRY
4464 00004946 50                <1>       PUSH   eAX                ; SAVE NUMBER TRANSFERRED
4465 00004947 E884FEFFFF        <1>       CALL   XLAT_OLD           ; TRANSLATE STATE TO COMPATIBLE MODE
4466 0000494C 58                <1>       POP    eAX                ; RESTORE NUMBER TRANSFERRED
4467 0000494D E8A7030000        <1>       CALL   SETUP_END          ; VARIOUS CLEANUPS
4468 00004952 C3                <1>       RETn
4469                            <1>
4470                            <1> ;--------------------------------------------------------------------------
4471                            <1> ; SETUP_STATE:     INITIALIZES START AND END RATES.
4472                            <1> ;--------------------------------------------------------------------------
4473                            <1> SETUP_STATE:
4474 00004953 F687[3D810100]10  <1>       TEST   byte [DSK_STATE+eDI], MED_DET ; MEDIA DETERMINED ?
4475 0000495A 7537              <1>       JNZ    short J1C          ; NO STATES IF DETERMINED
4476 0000495C 66B84000          <1>       MOV   AX,(RATE_500*256)+RATE_300  ; AH = START RATE, AL = END RATE
4477 00004960 F687[3D810100]04  <1>       TEST   byte [DSK_STATE+eDI],DRV_DET ; DRIVE ?
4478 00004967 740D              <1>       JZ     short AX_SET       ; DO NOT KNOW DRIVE
4479 00004969 F687[3D810100]02  <1>       TEST   byte [DSK_STATE+eDI], FMT_CAPA ; MULTI-RATE?
4480 00004970 7504              <1>       JNZ    short AX_SET       ; JUMP IF YES
4481 00004972 66B88080          <1>       MOV   AX,RATE_250*257     ; START A END RATE 250 FOR 360 DRIVE
4482                            <1> AX_SET:
4483 00004976 80A7[3D810100]1F  <1>       AND    byte [DSK_STATE+eDI], ~(RATE_MSK+DBL_STEP) ; TURN OFF THE RATE
4484 0000497D 08A7[3D810100]    <1>       OR     [DSK_STATE+eDI], AH ; RATE FIRST TO TRY
4485 00004983 8025[38810100]F3  <1>       AND    byte [LASTRATE], ~STRT_MSK ; ERASE LAST TO TRY RATE BITS
4486 0000498A C0C804            <1>       ROR    AL,4               ; TO OPERATION LAST RATE LOCATION
4487 0000498D 0805[38810100]    <1>       OR     [LASTRATE], AL     ; LAST RATE
4488                            <1> J1C:
4489 00004993 C3                <1>       RETn
4490                            <1>
4491                            <1> ;--------------------------------------------------------------------------
4492                            <1> ;  FMT_INIT: ESTABLISH STATE IF UNESTABLISHED AT FORMAT TIME.
4493                            <1> ;--------------------------------------------------------------------------
4494                            <1> FMT_INIT:
4495 00004994 F687[3D810100]10  <1>       TEST   byte [DSK_STATE+eDI], MED_DET ; IS MEDIA ESTABLISHED
4496 0000499B 7546              <1>       JNZ    short F1_OUT       ; IF SO RETURN
4497 0000499D E81C040000        <1>       CALL   CMOS_TYPE          ; RETURN DRIVE TYPE IN AL
4498                            <1>       ;; 20/02/2015
4499                            <1>       ;;JC   short CL_DRV       ; ERROR IN CMOS ASSUME NO DRIVE
4500 000049A2 7440              <1>       jz     short CL_DRV ;; 20/02/2015
4501 000049A4 FEC8              <1>       DEC    AL                 ; MAKE ZERO ORIGIN
4502                            <1>       ;;JS   short CL_DRV       ; NO DRIVE IF AL 0
4503 000049A6 8AA7[3D810100]    <1>       MOV    AH, [DSK_STATE+eDI] ; AH = CURRENT STATE
4504 000049AC 80E40F            <1>       AND    AH, ~(MED_DET+DBL_STEP+RATE_MSK) ; CLEAR
4505 000049AF 08C0              <1>       OR     AL,AL              ; CHECK FOR 360
4506 000049B1 7505              <1>       JNZ    short N_360        ; IF 360 WILL BE 0
4507 000049B3 80CC90            <1>       OR     AH,MED_DET+RATE_250 ; ESTABLISH MEDIA
4508 000049B6 EB25              <1>       JMP    SHORT SKP_STATE    ; SKIP OTHER STATE PROCESSING
4509                            <1> N_360:
4510 000049B8 FEC8              <1>       DEC    AL                 ; 1.2 M DRIVE
4511 000049BA 7505              <1>       JNZ    short N_12         ; JUMP IF NOT
4512                            <1> F1_RATE:
4513 000049BC 80CC10            <1>       OR     AH,MED_DET+RATE_500 ; SET FORMAT RATE
4514 000049BF EB1C              <1>       JMP    SHORT SKP_STATE    ; SKIP OTHER STATE PROCESSING
4515                            <1> N_12:
4516 000049C1 FEC8              <1>       DEC    AL                 ; CHECK FOR TYPE 3
4517 000049C3 750F              <1>       JNZ    short N_720        ; JUMP IF NOT
4518 000049C5 F6C404            <1>       TEST   AH,DRV_DET         ; IS DRIVE DETERMINED
4519 000049C8 7410              <1>       JZ     short ISNT_12      ; TREAT AS NON 1.2 DRIVE
4520 000049CA F6C402            <1>       TEST   AH,FMT_CAPA        ; IS 1.2M
4521 000049CD 740B              <1>       JZ     short ISNT_12      ; JUMP IF NOT
4522 000049CF 80CC50            <1>       OR     AH,MED_DET+RATE_300 ; RATE 300
4523 000049D2 EB09              <1>       JMP    SHORT SKP_STATE    ; CONTINUE
4524                            <1> N_720:
4525 000049D4 FEC8              <1>       DEC    AL                 ; CHECK FOR TYPE 4
4526 000049D6 750C              <1>       JNZ    short CL_DRV       ; NO DRIVE, CMOS BAD
4527 000049D8 EBE2              <1>       JMP    SHORT F1_RATE
4528                            <1> ISNT_12:
4529 000049DA 80CC90            <1>       OR     AH,MED_DET+RATE_250 ; MUST BE RATE 250
4530                            <1>
4531                            <1> SKP_STATE:
4532 000049DD 88A7[3D810100]    <1>       MOV    [DSK_STATE+eDI], AH ; STORE AWAY
4533                            <1> F1_OUT:
4534 000049E3 C3                <1>       RETn
4535                            <1> CL_DRV:
4536 000049E4 30E4              <1>       XOR    AH,AH              ; CLEAR STATE
4537 000049E6 EBF5              <1>       JMP    SHORT SKP_STATE    ; SAVE IT
4538                            <1>
4539                            <1> ;--------------------------------------------------------------------------
4540                            <1> ; MED_CHANGE
```

```
4541                            <1> ;     CHECKS FOR MEDIA CHANGE, RESETS MEDIA CHANGE,
4542                            <1> ;     CHECKS MEDIA CHANGE AGAIN.
4543                            <1> ;
4544                            <1> ; ON EXIT:  CY = 1 MEANS MEDIA CHANGE OR TIMEOUT
4545                            <1> ;           @DSKETTE_STATUS = ERROR CODE
4546                            <1> ;--------------------------------------------------------------------------------
4547                            <1> MED_CHANGE:
4548 000049E8 E876060000       <1>       CALL  READ_DSKCHNG        ; READ DISK CHANCE LINE STATE
4549 000049ED 7447              <1>       JZ    short MC_OUT        ; BYPASS HANDLING DISK CHANGE LINE
4550 000049EF 80A7[3D810100]EF  <1>       AND   byte [DSK_STATE+eDI], ~MED_DET ; CLEAR STATE FOR THIS DRIVE
4551                            <1>
4552                            <1> ;     THIS SEQUENCE ENSURES WHENEVER A DISKETTE IS CHANGED THAT
4553                            <1> ;     ON THE NEXT OPERATION THE REQUIRED MOTOR START UP TIME WILL
4554                            <1> ;     BE WAITED. (DRIVE MOTOR MAY GO OFF UPON DOOR OPENING).
4555                            <1>
4556 000049F6 6689F9            <1>       MOV   CX,DI               ; CL = DRIVE 0
4557 000049F9 B001              <1>       MOV   AL,1                ; MOTOR ON BIT MASK
4558 000049FB D2E0              <1>       SHL   AL,CL               ; TO APPROPRIATE POSITION
4559 000049FD F6D0              <1>       NOT   AL                  ; KEEP ALL BUT MOTOR ON
4560 000049FF FA                <1>       CLI                       ; NO INTERRUPTS
4561 00004A00 2005[2E810100]    <1>       AND   [MOTOR_STATUS], AL  ; TURN MOTOR OFF INDICATOR
4562 00004A06 FB                <1>       STI                       ; INTERRUPTS ENABLED
4563 00004A07 E800040000        <1>       CALL  MOTOR_ON            ; TURN MOTOR ON
4564                            <1>
4565                            <1> ;-----      THIS SEQUENCE OF SEEKS IS USED TO RESET DISKETTE CHANGE SIGNAL
4566                            <1>
4567 00004A0C E876F9FFFF        <1>       CALL  DSK_RESET           ; RESET NEC
4568 00004A11 B501              <1>       MOV   CH,01H              ; MOVE TO CYLINDER 1
4569 00004A13 E8EF040000        <1>       CALL  SEEK                ; ISSUE SEEK
4570 00004A18 30ED              <1>       XOR   CH,CH               ; MOVE TO CYLINDER 0
4571 00004A1A E8E8040000        <1>       CALL  SEEK                ; ISSUE SEEK
4572 00004A1F C605[30810100]06  <1>       MOV   byte [DSKETTE_STATUS], MEDIA_CHANGE ; STORE IN STATUS
4573                            <1> OK1:
4574 00004A26 E838060000        <1>       CALL  READ_DSKCHNG        ; CHECK MEDIA CHANGED AGAIN
4575 00004A2B 7407              <1>       JZ    short OK2           ; IF ACTIVE, NO DISKETTE, TIMEOUT
4576                            <1> OK4:
4577 00004A2D C605[30810100]80  <1>       MOV   byte [DSKETTE_STATUS], TIME_OUT ; TIMEOUT IF DRIVE EMPTY
4578                            <1> OK2:
4579 00004A34 F9                <1>       STC                       ; MEDIA CHANGED, SET CY
4580 00004A35 C3                <1>       RETn
4581                            <1> MC_OUT:
4582 00004A36 F8                <1>       CLC                       ; NO MEDIA CHANGED, CLEAR CY
4583 00004A37 C3                <1>       RETn
4584                            <1>
4585                            <1> ;--------------------------------------------------------------------------------
4586                            <1> ; SEND_RATE
4587                            <1> ;     SENDS DATA RATE COMMAND TO NEC
4588                            <1> ; ON ENTRY:  DI = DRIVE #
4589                            <1> ; ON EXIT:   NONE
4590                            <1> ; REGISTERS ALTERED: DX
4591                            <1> ;--------------------------------------------------------------------------------
4592                            <1> SEND_RATE:
4593                            <1>       ;PUSH  AX                 ; SAVE REG.
4594                            <1>       ; 11/04/2021
4595 00004A38 50                <1>       push  eax
4596 00004A39 8025[38810100]3F  <1>       AND   byte [LASTRATE], ~SEND_MSK ; ELSE CLEAR LAST RATE ATTEMPTED
4597 00004A40 8A87[3D810100]    <1>       MOV   AL, [DSK_STATE+eDI] ; GET RATE STATE OF THIS DRIVE
4598 00004A46 24C0              <1>       AND   AL,SEND_MSK         ; KEEP ONLY RATE BITS
4599 00004A48 0805[38810100]    <1>       OR    [LASTRATE], AL            ; SAVE NEW RATE FOR NEXT CHECK
4600 00004A4E C0C002            <1>       ROL   AL,2                ; MOVE TO BIT OUTPUT POSITIONS
4601 00004A51 66BAF703          <1>       MOV   DX,03F7H            ; OUTPUT NEW DATA RATE
4602 00004A55 EE                <1>       OUT   DX,AL
4603                            <1>       ;POP   AX                 ; RESTORE REG.
4604                            <1>       ; 11/04/2021
4605 00004A56 58                <1>       pop   eax
4606 00004A57 C3                <1>       RETn
4607                            <1>
4608                            <1> ;--------------------------------------------------------------------------------
4609                            <1> ; CHK_LASTRATE
4610                            <1> ;     CHECK PREVIOUS DATE RATE SNT TO THE CONTROLLER.
4611                            <1> ; ON ENTRY:
4612                            <1> ;     DI = DRIVE #
4613                            <1> ; ON EXIT:
4614                            <1> ;     ZF =  1 DATA RATE IS THE SAME AS THE LAST RATE SENT TO NEC
4615                            <1> ;     ZF =  0 DATA RATE IS DIFFERENT FROM LAST RATE
4616                            <1> ; REGISTERS ALTERED: DX
4617                            <1> ;--------------------------------------------------------------------------------
4618                            <1> CHK_LASTRATE:
4619                            <1>       ;PUSH  AX                 ; SAVE REG.
4620                            <1>       ; 11/04/2021
4621 00004A58 50                <1>       push  eax
4622 00004A59 2225[38810100]    <1>       AND   AH, [LASTRATE]      ; GET LAST DATA RATE SELECTED
4623 00004A5F 8A87[3D810100]    <1>       MOV   AL, [DSK_STATE+eDI] ; GET RATE STATE OF THIS DRIVE
4624 00004A65 6625C0C0          <1>         AND     AX, SEND_MSK*257  ; KEEP ONLY RATE BITS OF BOTH
4625 00004A69 38E0              <1>       CMP   AL, AH              ; COMPARE TO PREVIOUSLY TRIED
4626                            <1>                                 ; ZF = 1 RATE IS THE SAME
4627                            <1>       ;POP   AX                 ; RESTORE REG.
4628                            <1>       ; 11/04/2021
4629 00004A6B 58                <1>       pop   eax
4630 00004A6C C3                <1>       RETn
4631                            <1>
4632                            <1> ;--------------------------------------------------------------------------------
4633                            <1> ; DMA_SETUP
4634                            <1> ;     THIS ROUTINE SETS UP THE DMA FOR READ/WRITE/VERIFY OPERATIONS.
4635                            <1> ;
4636                            <1> ; ON ENTRY:  AL = DMA COMMAND
4637                            <1> ;
4638                            <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
4639                            <1> ;--------------------------------------------------------------------------------
4640                            <1>
4641                            <1> ; SI = Head #, # of Sectors or DASD Type
4642                            <1>
4643                            <1> ; 22/08/2015
4644                            <1> ; 08/02/2015 - Protected Mode Modification
4645                            <1> ; 06/02/2015 - 07/02/2015
```

```
                        <1> ; NOTE: Buffer address must be in 1st 16MB of Physical Memory (24 bit limit).
                        <1> ; (DMA Addres = Physical Address)
                        <1> ; (Retro UNIX 386 v1 Kernel/System Mode Virtual Address = Physical Address)
                        <1> ;
                        <1>
                        <1>
                        <1> ; 04/02/2016 (clc)
                        <1> ; 20/02/2015 modification (source: AWARD BIOS 1999, DMA_SETUP)
                        <1> ; 16/12/2014 (IODELAY)
                        <1>
                        <1> DMA_SETUP:
                        <1>
                        <1> ;; 20/02/2015
00004A6D 8B5504         <1>         mov    edx, [ebp+4]        ; Buffer address
00004A70 F7C2000000FF   <1>         test   edx, 0FF000000h     ; 16 MB limit (22/08/2015, bugfix)
00004A76 756C           <1>         jnz    short dma_bnd_err_stc
                        <1>         ;
                        <1>         ;push  ax                  ; DMA command
                        <1>         ; 11/04/2021
00004A78 50             <1>         push   eax
00004A79 52             <1>         push   edx                 ; *
00004A7A B203           <1>         mov    dl, 3               ; GET BYTES/SECTOR PARAMETER
00004A7C E846030000     <1>         call   GET_PARM            ;
00004A81 88E1           <1>         mov    cl, ah              ; SHIFT COUNT (0=128, 1=256, 2=512 ETC)
00004A83 6689F0         <1>         mov    ax, si              ; Sector count
00004A86 88C4           <1>         mov    ah, al              ; AH =  # OF SECTORS
00004A88 28C0           <1>         sub    al, al              ; AL = 0, AX = # SECTORS * 256
00004A8A 66D1E8         <1>         shr    ax, 1               ; AX = # SECTORS * 128
00004A8D 66D3E0         <1>         shl    ax, cl              ; SHIFT BY PARAMETER VALUE
00004A90 6648           <1>         dec    ax                  ; -1 FOR DMA VALUE
00004A92 6689C1         <1>         mov    cx, ax
00004A95 5A             <1>         pop    edx                 ; *
                        <1>         ;pop   ax
                        <1>         ; 11/04/2021
00004A96 58             <1>         pop    eax
00004A97 3C42           <1>         cmp    al, 42h
00004A99 7507           <1>           jne     short NOT_VERF
00004A9B BA0000FF00     <1>         mov    edx, 0FF0000h
00004AA0 EB08           <1>         jmp    short J33
                        <1> NOT_VERF:
00004AA2 6601CA         <1>         add    dx, cx              ; check for overflow
00004AA5 723E           <1>         jc     short dma_bnd_err
                        <1>         ;
00004AA7 6629CA         <1>         sub    dx, cx              ; Restore start address
                        <1> J33:
00004AAA FA             <1>         CLI                        ; DISABLE INTERRUPTS DURING DMA SET-UP
00004AAB E60C           <1>         OUT    DMA+12,AL           ; SET THE FIRST/LA5T F/F
                        <1>         IODELAY                    ; WAIT FOR I/O
2990 00004AAD EB00      <2>  jmp short $+2
2991 00004AAF EB00      <2>  jmp short $+2
00004AB1 E60B           <1>         OUT    DMA+11,AL           ; OUTPUT THE MODE BYTE
00004AB3 89D0           <1>         mov    eax, edx            ; Buffer address
00004AB5 E604           <1>         OUT    DMA+4,AL            ; OUTPUT LOW ADDRESS
                        <1>         IODELAY                    ; WAIT FOR I/O
2990 00004AB7 EB00      <2>  jmp short $+2
2991 00004AB9 EB00      <2>  jmp short $+2
00004ABB 88E0           <1>         MOV    AL,AH
00004ABD E604           <1>         OUT    DMA+4,AL            ; OUTPUT HIGH ADDRESS
00004ABF C1E810         <1>         shr    eax, 16
                        <1>         IODELAY                    ; I/O WAIT STATE
2990 00004AC2 EB00      <2>  jmp short $+2
2991 00004AC4 EB00      <2>  jmp short $+2
00004AC6 E681           <1>         OUT    081H,AL             ; OUTPUT highest BITS TO PAGE REGISTER
                        <1>         IODELAY
2990 00004AC8 EB00      <2>  jmp short $+2
2991 00004ACA EB00      <2>  jmp short $+2
00004ACC 6689C8         <1>         mov    ax, cx              ; Byte count - 1
00004ACF E605           <1>         OUT    DMA+5,AL            ; LOW BYTE OF COUNT
                        <1>         IODELAY                    ; WAIT FOR I/O
2990 00004AD1 EB00      <2>  jmp short $+2
2991 00004AD3 EB00      <2>  jmp short $+2
00004AD5 88E0           <1>         MOV    AL, AH
00004AD7 E605           <1>         OUT    DMA+5,AL            ; HIGH BYTE OF COUNT
                        <1>         IODELAY
2990 00004AD9 EB00      <2>  jmp short $+2
2991 00004ADB EB00      <2>  jmp short $+2
00004ADD FB             <1>         STI                        ; RE-ENABLE INTERRUPTS
00004ADE B002           <1>         MOV    AL, 2               ; MODE FOR 8237
00004AE0 E60A           <1>         OUT    DMA+10,AL           ; INITIALIZE THE DISKETTE CHANNEL
                        <1>
00004AE2 F8             <1>         clc    ; 04/02/2016
00004AE3 C3             <1>         retn
                        <1>
                        <1> dma_bnd_err_stc:
00004AE4 F9             <1>         stc
                        <1> dma_bnd_err:
00004AE5 C605[30810100]09 <1>       MOV    byte [DSKETTE_STATUS], DMA_BOUNDARY ; SET ERROR
00004AEC C3             <1>         RETn                       ; CY SET BY ABOVE IF ERROR
                        <1>
                        <1> ;; 16/12/2014
                        <1> ;;    CLI                          ; DISABLE INTERRUPTS DURING DMA SET-UP
                        <1> ;;    OUT    DMA+12,AL             ; SET THE FIRST/LA5T F/F
                        <1> ;;    ;JMP   $+2                   ; WAIT FOR I/O
                        <1> ;;    IODELAY
                        <1> ;;    OUT    DMA+11,AL             ; OUTPUT THE MODE BYTE
                        <1> ;;    ;SIODELAY
                        <1> ;;    ;CMP   AL, 42H               ; DMA VERIFY COMMAND
                        <1> ;;    ;JNE   short NOT_VERF        ; NO
                        <1> ;;    ;XOR   AX, AX                ; START ADDRESS
                        <1> ;;    ;JMP   SHORT J33
                        <1> ;;;NOT_VERF:
                        <1> ;;    ;MOV   AX,ES                 ; GET THE ES VALUE
                        <1> ;;    ;ROL   AX,4                  ; ROTATE LEFT
                        <1> ;;    ;MOV   CH,AL                 ; GET HIGHEST NIBBLE OF ES TO CH
                        <1> ;;    ;AND   AL,11110000B          ; ZERO THE LOW NIBBLE FROM SEGMENT
```

```
                        <1> ;;    ;ADD   AX,[BP+2]          ; TEST FOR CARRY FROM ADDITION
                        <1> ;;    mov    eax, [ebp+4] ; 06/02/2015
                        <1> ;;    ;JNC   short J33
                        <1> ;;    ;INC   CH                 ; CARRY MEANS HIGH 4 BITS MUST BE INC
                        <1> ;;;J33:
                        <1> ;;    PUSH   eAX                ; SAVE START ADDRESS
                        <1> ;;    OUT    DMA+4,AL           ; OUTPUT LOW ADDRESS
                        <1> ;;    ;JMP   $+2                ; WAIT FOR I/O
                        <1> ;;    IODELAY
                        <1> ;;    MOV    AL,AH
                        <1> ;;    OUT    DMA+4,AL           ; OUTPUT HIGH ADDRESS
                        <1> ;;    shr    eax, 16           ; 07/02/2015
                        <1> ;;    ;MOV   AL,CH              ; GET HIGH 4 BITS
                        <1> ;;    ;JMP   $+2                ; I/O WAIT STATE
                        <1> ;;    IODELAY
                        <1> ;;    ;AND   AL,00001111B
                        <1> ;;    OUT    081H,AL            ; OUTPUT HIGH 4 BITS TO PAGE REGISTER
                        <1> ;;    ;SIODELAY
                        <1> ;;
                        <1> ;;;----- DETERMINE COUNT
                        <1> ;;    sub    eax, eax ; 08/02/2015
                        <1> ;;    MOV    AX, SI             ; AL =  # OF SECTORS
                        <1> ;;    XCHG   AL, AH             ; AH =  # OF SECTORS
                        <1> ;;    SUB    AL, AL             ; AL = 0, AX = # SECTORS * 256
                        <1> ;;    SHR    AX, 1              ; AX = # SECTORS * 128
                        <1> ;;    PUSH   AX                 ; SAVE # OF SECTORS * 128
                        <1> ;;    MOV    DL, 3              ; GET BYTES/SECTOR PARAMETER
                        <1> ;;    CALL   GET_PARM           ; "
                        <1> ;;    MOV    CL,AH              ; SHIFT COUNT (0=128, 1=256, 2=512 ETC)
                        <1> ;;    POP    AX                 ; AX = # SECTORS * 128
                        <1> ;;    SHL    AX,CL              ; SHIFT BY PARAMETER VALUE
                        <1> ;;    DEC    AX                 ; -1 FOR DMA VALUE
                        <1> ;;    PUSH   eAX  ; 08/02/2015  ; SAVE COUNT VALUE
                        <1> ;;    OUT    DMA+5,AL           ; LOW BYTE OF COUNT
                        <1> ;;    ;JMP   $+2                ; WAIT FOR I/O
                        <1> ;;    IODELAY
                        <1> ;;    MOV    AL, AH
                        <1> ;;    OUT    DMA+5,AL           ; HIGH BYTE OF COUNT
                        <1> ;;    ;IODELAY
                        <1> ;;    STI                       ; RE-ENABLE INTERRUPTS
                        <1> ;;    POP    eCX  ; 08/02/2015  ; RECOVER COUNT VALUE
                        <1> ;;    POP    eAX  ; 08/02/2015  ; RECOVER ADDRESS VALUE
                        <1> ;;    ;ADD   AX, CX             ; ADD, TEST FOR 64K OVERFLOW
                        <1> ;;    add    ecx, eax ; 08/02/2015
                        <1> ;;    MOV    AL, 2              ; MODE FOR 8237
                        <1> ;;    ;JMP   $+2                ; WAIT FOR I/O
                        <1> ;;    SIODELAY
                        <1> ;;    OUT    DMA+10, AL         ; INITIALIZE THE DISKETTE CHANNEL
                        <1> ;;    ;JNC   short NO_BAD       ; CHECK FOR ERROR
                        <1> ;;    jc     short dma_bnd_err ; 08/02/2015
                        <1> ;;    and    ecx, 0FFF00000h ; 16 MB limit
                        <1> ;;    jz     short NO_BAD
                        <1> ;;dma_bnd_err:
                        <1> ;;    MOV    byte [DSKETTE_STATUS], DMA_BOUNDARY ; SET ERROR
                        <1> ;;NO_BAD:
                        <1> ;;    RETn                      ; CY SET BY ABOVE IF ERROR
                        <1>
                        <1> ;--------------------------------------------------------------------------------
                        <1> ; FMTDMA_SET
                        <1> ;     THIS ROUTINE SETS UP THE DMA CONTROLLER FOR A FORMAT OPERATION.
                        <1> ;
                        <1> ; ON ENTRY:  NOTHING REQUIRED
                        <1> ;
                        <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
                        <1> ;--------------------------------------------------------------------------------
                        <1>
                        <1> FMTDMA_SET:
                        <1> ;; 20/02/2015 modification
00004AED 8B5504         <1>       mov    edx, [ebp+4]       ; Buffer address
00004AF0 F7C20000F0FF   <1>       test   edx, 0FFF00000h    ; 16 MB limit
00004AF6 75EC           <1>       jnz    short dma_bnd_err_stc
                        <1>       ;
                        <1>       ;push  dx                 ; *
                        <1>       ; 11/04/2021
00004AF8 52             <1>       push   edx
00004AF9 B204           <1>       mov    DL, 4              ; SECTORS/TRACK VALUE IN PARM TABLE
00004AFB E8C7020000     <1>       call   GET_PARM           ; "
00004B00 88E0           <1>       mov    al, ah             ; AL = SECTORS/TRACK VALUE
00004B02 28E4           <1>       sub    ah, ah             ; AX = SECTORS/TRACK VALUE
00004B04 66C1E002       <1>       shl    ax, 2              ; AX = SEC/TRK * 4 (OFFSET C,H,R,N)
00004B08 6648           <1>       dec    ax                 ; -1 FOR DMA VALUE
00004B0A 6689C1         <1>       mov    cx, ax
                        <1>       ;pop   dx                 ; *
                        <1>       ; 11/04/2021
00004B0D 5A             <1>       pop    edx
00004B0E 6601CA         <1>       add    dx, cx             ; check for overflow
00004B11 72D2           <1>       jc     short dma_bnd_err
                        <1>       ;
00004B13 6629CA         <1>       sub    dx, cx             ; Restore start address
                        <1>       ;
00004B16 B04A           <1>       MOV    AL, 04AH           ; WILL WRITE TO THE DISKETTE
00004B18 FA             <1>       CLI                       ; DISABLE INTERRUPTS DURING DMA SET-UP
00004B19 E60C           <1>       OUT    DMA+12,AL          ; SET THE FIRST/LA5T F/F
                        <1>       IODELAY                   ; WAIT FOR I/O
2990 00004B1B EB00      <2>  jmp short $+2
2991 00004B1D EB00      <2>  jmp short $+2
00004B1F E60B           <1>       OUT    DMA+11,AL          ; OUTPUT THE MODE BYTE
00004B21 89D0           <1>       mov    eax, edx           ; Buffer address
00004B23 E604           <1>       OUT    DMA+4,AL           ; OUTPUT LOW ADDRESS
                        <1>       IODELAY                   ; WAIT FOR I/O
2990 00004B25 EB00      <2>  jmp short $+2
2991 00004B27 EB00      <2>  jmp short $+2
00004B29 88E0           <1>       MOV    AL,AH
00004B2B E604           <1>       OUT    DMA+4,AL           ; OUTPUT HIGH ADDRESS
00004B2D C1E810         <1>       shr    eax, 16
```

```
4840                                    <1>       IODELAY                   ; I/O WAIT STATE
2990 00004B30 EB00                      <2>  jmp short $+2
2991 00004B32 EB00                      <2>  jmp short $+2
4841 00004B34 E681                      <1>       OUT    081H,AL            ; OUTPUT highest BITS TO PAGE REGISTER
4842                                    <1>       IODELAY
2990 00004B36 EB00                      <2>  jmp short $+2
2991 00004B38 EB00                      <2>  jmp short $+2
4843 00004B3A 6689C8                    <1>       mov    ax, cx             ; Byte count - 1
4844 00004B3D E605                      <1>       OUT    DMA+5,AL           ; LOW BYTE OF COUNT
4845                                    <1>       IODELAY                   ; WAIT FOR I/O
2990 00004B3F EB00                      <2>  jmp short $+2
2991 00004B41 EB00                      <2>  jmp short $+2
4846 00004B43 88E0                      <1>       MOV    AL, AH
4847 00004B45 E605                      <1>       OUT    DMA+5,AL           ; HIGH BYTE OF COUNT
4848                                    <1>       IODELAY
2990 00004B47 EB00                      <2>  jmp short $+2
2991 00004B49 EB00                      <2>  jmp short $+2
4849 00004B4B FB                        <1>       STI                       ; RE-ENABLE INTERRUPTS
4850 00004B4C B002                      <1>       MOV    AL, 2              ; MODE FOR 8237
4851 00004B4E E60A                      <1>       OUT    DMA+10,AL          ; INITIALIZE THE DISKETTE CHANNEL
4852 00004B50 C3                        <1>       retn
4853                                    <1>
4854                                    <1> ;; 08/02/2015 - Protected Mode Modification
4855                                    <1> ;;    MOV    AL, 04AH           ; WILL WRITE TO THE DISKETTE
4856                                    <1> ;;    CLI                       ; DISABLE INTERRUPTS DURING DMA SET-UP
4857                                    <1> ;;    OUT    DMA+12,AL          ; SET THE FIRST/LA5T F/F
4858                                    <1> ;;    ;JMP   $+2                ; WAIT FOR I/O
4859                                    <1> ;;    IODELAY
4860                                    <1> ;;    OUT    DMA+11,AL          ; OUTPUT THE MODE BYTE
4861                                    <1> ;;    ;MOV   AX,ES              ; GET THE ES VALUE
4862                                    <1> ;;    ;ROL   AX,4               ; ROTATE LEFT
4863                                    <1> ;;    ;MOV   CH,AL              ; GET HIGHEST NIBBLE OF ES TO CH
4864                                    <1> ;;    ;AND   AL,11110000B       ; ZERO THE LOW NIBBLE FROM SEGMENT
4865                                    <1> ;;    ;ADD   AX,[BP+2]          ; TEST FOR CARRY FROM ADDITION
4866                                    <1> ;;    ;JNC   short J33A
4867                                    <1> ;;    ;INC   CH                 ; CARRY MEANS HIGH 4 BITS MUST BE INC
4868                                    <1> ;;    mov    eax, [ebp+4] ; 08/02/2015
4869                                    <1> ;;;J33A:
4870                                    <1> ;;    PUSH   eAX ; 08/02/2015   ; SAVE START ADDRESS
4871                                    <1> ;;    OUT    DMA+4,AL           ; OUTPUT LOW ADDRESS
4872                                    <1> ;;    ;JMP   $+2                ; WAIT FOR I/O
4873                                    <1> ;;    IODELAY
4874                                    <1> ;;    MOV    AL,AH
4875                                    <1> ;;    OUT    DMA+4,AL           ; OUTPUT HIGH ADDRESS
4876                                    <1> ;;    shr    eax, 16 ; 08/02/2015
4877                                    <1> ;;    ;MOV   AL,CH              ; GET HIGH 4 BITS
4878                                    <1> ;;    ;JMP   $+2                ; I/O WAIT STATE
4879                                    <1> ;;    IODELAY
4880                                    <1> ;;    ;AND   AL,00001111B
4881                                    <1> ;;    OUT    081H,AL            ; OUTPUT HIGH 4 BITS TO PAGE REGISTER
4882                                    <1> ;;
4883                                    <1> ;;;----- DETERMINE COUNT
4884                                    <1> ;;    sub    eax, eax ; 08/02/2015
4885                                    <1> ;;    MOV    DL, 4              ; SECTORS/TRACK VALUE IN PARM TABLE
4886                                    <1> ;;    CALL   GET_PARM           ; "
4887                                    <1> ;;    XCHG   AL, AH             ; AL = SECTORS/TRACK VALUE
4888                                    <1> ;;    SUB    AH, AH             ; AX = SECTORS/TRACK VALUE
4889                                    <1> ;;    SHL    AX, 2              ; AX = SEC/TRK * 4 (OFFSET C,H,R,N)
4890                                    <1> ;;    DEC    AX                 ; -1 FOR DMA VALUE
4891                                    <1> ;;    PUSH   eAX    ; 08/02/2015 ; SAVE # OF BYTES TO BE TRANSFERED
4892                                    <1> ;;    OUT    DMA+5,AL           ; LOW BYTE OF COUNT
4893                                    <1> ;;    ;JMP   $+2                ; WAIT FOR I/O
4894                                    <1> ;;    IODELAY
4895                                    <1> ;;    MOV    AL, AH
4896                                    <1> ;;    OUT    DMA+5,AL           ; HIGH BYTE OF COUNT
4897                                    <1> ;;    STI                       ; RE-ENABLE INTERRUPTS
4898                                    <1> ;;    POP    eCX    ; 08/02/2015 ; RECOVER COUNT VALUE
4899                                    <1> ;;    POP    eAX    ; 08/02/2015 ; RECOVER ADDRESS VALUE
4900                                    <1> ;;    ;ADD   AX, CX             ; ADD, TEST FOR 64K OVERFLOW
4901                                    <1> ;;    add    ecx, eax ; 08/02/2015
4902                                    <1> ;;    MOV    AL, 2              ; MODE FOR 8237
4903                                    <1> ;;    ;JMP   $+2                ; WAIT FOR I/O
4904                                    <1> ;;    SIODELAY
4905                                    <1> ;;    OUT    DMA+10, AL         ; INITIALIZE THE DISKETTE CHANNEL
4906                                    <1> ;;    ;JNC   short FMTDMA_OK    ; CHECK FOR ERROR
4907                                    <1> ;;    jc     short fmtdma_bnd_err ; 08/02/2015
4908                                    <1> ;;    and    ecx, 0FFF00000h  ; 16 MB limit
4909                                    <1> ;;    jz     short FMTDMA_OK
4910                                    <1> ;;    stc    ; 20/02/2015
4911                                    <1> ;;fmtdma_bnd_err:
4912                                    <1> ;;    MOV    byte [DSKETTE_STATUS], DMA_BOUNDARY ; SET ERROR
4913                                    <1> ;;FMTDMA_OK:
4914                                    <1> ;;    RETn                      ; CY SET BY ABOVE IF ERROR
4915                                    <1>
4916                                    <1> ;--------------------------------------------------------------------------------
4917                                    <1> ; NEC_INIT
4918                                    <1> ;     THIS ROUTINE SEEKS TO THE REQUESTED TRACK AND INITIALIZES
4919                                    <1> ;     THE NEC FOR THE READ/WRITE/VERIFY/FORMAT OPERATION.
4920                                    <1> ;
4921                                    <1> ; ON ENTRY:  AH = NEC COMMAND TO BE PERFORMED
4922                                    <1> ;
4923                                    <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
4924                                    <1> ;--------------------------------------------------------------------------------
4925                                    <1> NEC_INIT:
4926                                    <1>       ;PUSH  AX                 ; SAVE NEC COMMAND
4927                                    <1>       ; 11/04/2021
4928 00004B51 50                        <1>       push   eax
4929 00004B52 E8B5020000                <1>       CALL   MOTOR_ON           ; TURN MOTOR ON FOR SPECIFIC DRIVE
4930                                    <1>
4931                                    <1> ;-----      DO THE SEEK OPERATION
4932                                    <1>
4933 00004B57 8A6D01                    <1>       MOV    CH,[eBP+1]         ; CH = TRACK #
4934 00004B5A E8A8030000                <1>       CALL   SEEK               ; MOVE TO CORRECT TRACK
4935                                    <1>       ;POP   AX                 ; RECOVER COMMAND
4936                                    <1>       ; 11/04/2021
```

```
4937 00004B5F 58                <1>       pop    eax
4938 00004B60 721E              <1>       JC     short ER_1          ; ERROR ON SEEK
4939 00004B62 BB[804B0000]      <1>       MOV    eBX, ER_1           ; LOAD ERROR ADDRESS
4940 00004B67 53                <1>       PUSH   eBX                 ; PUSH NEC_OUT ERROR RETURN
4941                            <1>
4942                            <1> ;-----      SEND OUT THE PARAMETERS TO THE CONTROLLER
4943                            <1>
4944 00004B68 E860030000        <1>       CALL   NEC_OUTPUT          ; OUTPUT THE OPERATION COMMAND
4945 00004B6D 6689F0            <1>       MOV    AX,SI               ; AH = HEAD #
4946 00004B70 89FB              <1>       MOV    eBX,eDI                   ; BL = DRIVE #
4947 00004B72 C0E402            <1>       SAL    AH,2                ; MOVE IT TO BIT 2
4948 00004B75 80E404            <1>       AND    AH,00000100B        ; ISOLATE THAT BIT
4949 00004B78 08DC              <1>       OR     AH,BL               ; OR IN THE DRIVE NUMBER
4950 00004B7A E84E030000        <1>       CALL   NEC_OUTPUT          ; FALL THRU CY SET IF ERROR
4951 00004B7F 5B                <1>       POP    eBX                 ; THROW AWAY ERROR RETURN
4952                            <1> ER_1:
4953 00004B80 C3                <1>       RETn
4954                            <1>
4955                            <1> ;--------------------------------------------------------------------------------
4956                            <1> ; RWV_COM
4957                            <1> ;     THIS ROUTINE SENDS PARAMETERS TO THE NEC SPECIFIC TO THE
4958                            <1> ;     READ/WRITE/VERIFY OPERATIONS.
4959                            <1> ;
4960                            <1> ; ON ENTRY:  CS:BX = ADDRESS OF MEDIA/DRIVE PARAMETER TABLE
4961                            <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
4962                            <1> ;--------------------------------------------------------------------------------
4963                            <1> RWV_COM:
4964 00004B81 B8[CC4B0000]      <1>       MOV    eAX, ER_2           ; LOAD ERROR ADDRESS
4965 00004B86 50                <1>       PUSH   eAX                 ; PUSH NEC_OUT ERROR RETURN
4966 00004B87 8A6501            <1>       MOV    AH,[eBP+1]          ; OUTPUT TRACK #
4967 00004B8A E83E030000        <1>       CALL   NEC_OUTPUT
4968 00004B8F 6689F0            <1>       MOV    AX,SI               ; OUTPUT HEAD #
4969 00004B92 E836030000        <1>       CALL   NEC_OUTPUT
4970 00004B97 8A6500            <1>       MOV    AH,[eBP]            ; OUTPUT SECTOR #
4971 00004B9A E82E030000        <1>       CALL   NEC_OUTPUT
4972 00004B9F B203              <1>       MOV    DL,3                ; BYTES/SECTOR PARAMETER FROM BLOCK
4973 00004BA1 E821020000        <1>       CALL   GET_PARM            ; ... TO THE NEC
4974 00004BA6 E822030000        <1>       CALL   NEC_OUTPUT          ; OUTPUT TO CONTROLLER
4975 00004BAB B204              <1>       MOV    DL,4                ; EOT PARAMETER FROM BLOCK
4976 00004BAD E815020000        <1>       CALL   GET_PARM            ; ... TO THE NEC
4977 00004BB2 E816030000        <1>       CALL   NEC_OUTPUT          ; OUTPUT TO CONTROLLER
4978 00004BB7 8A6305            <1>       MOV    AH, [eBX+MD.GAP]    ; GET GAP LENGTH
4979                            <1> _R15:
4980 00004BBA E80E030000        <1>       CALL   NEC_OUTPUT
4981 00004BBF B206              <1>       MOV    DL,6                ; DTL PARAMETER PROM BLOCK
4982 00004BC1 E801020000        <1>       CALL   GET_PARM            ;  TO THE NEC
4983 00004BC6 E802030000        <1>       CALL   NEC_OUTPUT          ; OUTPUT TO CONTROLLER
4984 00004BCB 58                <1>       POP    eAX                 ; THROW AWAY ERROR EXIT
4985                            <1> ER_2:
4986 00004BCC C3                <1>       RETn
4987                            <1>
4988                            <1> ;--------------------------------------------------------------------------------
4989                            <1> ; NEC_TERM
4990                            <1> ;     THIS ROUTINE WAITS FOR THE OPERATION THEN ACCEPTS THE STATUS
4991                            <1> ;     FROM THE NEC FOR THE READ/WRITE/VERIFY/FORWAT OPERATION.
4992                            <1> ;
4993                            <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
4994                            <1> ;--------------------------------------------------------------------------------
4995                            <1> NEC_TERM:
4996                            <1>
4997                            <1> ;----- LET THE OPERATION HAPPEN
4998                            <1>
4999 00004BCD 56                <1>       PUSH   eSI                 ; SAVE HEAD #, # OF SECTORS
5000 00004BCE E805040000        <1>       CALL   WAIT_INT            ; WAIT FOR THE INTERRUPT
5001 00004BD3 9C                <1>       PUSHF
5002 00004BD4 E82F040000        <1>       CALL   RESULTS             ; GET THE NEC STATUS
5003 00004BD9 724B              <1>       JC     short SET_END_POP
5004 00004BDB 9D                <1>       POPF
5005 00004BDC 723E              <1>       JC     short SET_END       ; LOOK FOR ERROR
5006                            <1>
5007                            <1> ;----- CHECK THE RESULTS RETURNED BY THE CONTROLLER
5008                            <1>
5009 00004BDE FC                <1>       CLD                        ; SET THE CORRECT DIRECTION
5010 00004BDF BE[31810100]      <1>       MOV    eSI, NEC_STATUS     ; POINT TO STATUS FIELD
5011 00004BE4 AC                <1>       lodsb                      ; GET ST0
5012 00004BE5 24C0              <1>       AND    AL,11000000B        ; TEST FOR NORMAL TERMINATION
5013 00004BE7 7433              <1>       JZ     short SET_END
5014 00004BE9 3C40              <1>       CMP    AL,01000000B        ; TEST FOR ABNORMAL TERMINATION
5015 00004BEB 7527              <1>       JNZ    short J18           ; NOT ABNORMAL, BAD NEC
5016                            <1>
5017                            <1> ;----- ABNORMAL TERMINATION, FIND OUT WHY
5018                            <1>
5019 00004BED AC                <1>       lodsb                      ; GET ST1
5020 00004BEE D0E0              <1>       SAL    AL,1                ; TEST FOR EDT FOUND
5021 00004BF0 B404              <1>       MOV    AH,RECORD_NOT_FND
5022 00004BF2 7222              <1>       JC     short J19
5023 00004BF4 C0E002            <1>       SAL    AL,2
5024 00004BF7 B410              <1>       MOV    AH,BAD_CRC
5025 00004BF9 721B              <1>       JC     short J19
5026 00004BFB D0E0              <1>       SAL    AL,1                ; TEST FOR DMA OVERRUN
5027 00004BFD B408              <1>       MOV    AH,BAD_DMA
5028 00004BFF 7215              <1>       JC     short J19
5029 00004C01 C0E002            <1>       SAL    AL,2                ; TEST FOR RECORD NOT FOUND
5030 00004C04 B404              <1>       MOV    AH,RECORD_NOT_FND
5031 00004C06 720E              <1>       JC     short J19
5032 00004C08 D0E0              <1>       SAL    AL,1
5033 00004C0A B403              <1>       MOV    AH,WRITE_PROTECT    ; TEST FOR WRITE_PROTECT
5034 00004C0C 7208              <1>       JC     short J19
5035 00004C0E D0E0              <1>       SAL    AL,1                ; TEST MISSING ADDRESS MARK
5036 00004C10 B402              <1>       MOV    AH,BAD_ADDR_MARK
5037 00004C12 7202              <1>       JC     short J19
5038                            <1>
5039                            <1> ;----- NEC MUST HAVE FAILED
5040                            <1> J18:
5041 00004C14 B420              <1>       MOV    AH,BAD_NEC
```

```
                            <1> J19:
00004C16 0825[30810100]     <1>         OR      [DSKETTE_STATUS], AH
                            <1> SET_END:
00004C1C 803D[30810100]01   <1>         CMP     byte [DSKETTE_STATUS], 1 ; SET ERROR CONDITION
00004C23 F5                 <1>         CMC
00004C24 5E                 <1>         POP     eSI
00004C25 C3                 <1>         RETn                        ; RESTORE HEAD #, # OF SECTORS
                            <1>
                            <1> SET_END_POP:
00004C26 9D                 <1>         POPF
00004C27 EBF3               <1>         JMP     SHORT SET_END
                            <1>
                            <1> ;--------------------------------------------------------------------------
                            <1> ; DSTATE:   ESTABLISH STATE UPON SUCCESSFUL OPERATION.
                            <1> ;--------------------------------------------------------------------------
                            <1> DSTATE:
00004C29 803D[30810100]00   <1>         CMP     byte [DSKETTE_STATUS],0    ; CHECK FOR ERROR
00004C30 753E               <1>         JNZ     short SETBAC               ; IF ERROR JUMP
00004C32 808F[3D810100]10   <1>         OR      byte [DSK_STATE+eDI],MED_DET ; NO ERROR, MARK MEDIA AS DETERMINED
00004C39 F687[3D810100]04   <1>         TEST    byte [DSK_STATE+eDI],DRV_DET ; DRIVE DETERMINED ?
00004C40 752E               <1>         JNZ     short SETBAC        ; IF DETERMINED NO TRY TO DETERMINE
00004C42 8A87[3D810100]     <1>         MOV     AL,[DSK_STATE+eDI]  ; LOAD STATE
00004C48 24C0               <1>         AND     AL,RATE_MSK         ; KEEP ONLY RATE
00004C4A 3C80               <1>         CMP     AL,RATE_250         ; RATE 250 ?
00004C4C 751B               <1>         JNE     short M_12          ; NO, MUST BE 1.2M OR 1.44M DRIVE
                            <1>
                            <1> ;-----  CHECK IF IT IS 1.44M
                            <1>
00004C4E E86B010000         <1>         CALL    CMOS_TYPE           ; RETURN DRIVE TYPE IN (AL)
                            <1>         ;;20/02/2015
                            <1>         ;;JC    short M_12          ; CMOS BAD
00004C53 7414               <1>         jz      short M_12 ;; 20/02/2015
00004C55 3C04               <1>         CMP     AL, 4               ; 1.44MB DRIVE ?
00004C57 7410               <1>         JE      short M_12          ; YES
                            <1> M_720:
00004C59 80A7[3D810100]FD   <1>         AND     byte [DSK_STATE+eDI], ~FMT_CAPA ; TURN OFF FORMAT CAPABILITY
00004C60 808F[3D810100]04   <1>         OR      byte [DSK_STATE+eDI],DRV_DET  ; MARK DRIVE DETERMINED
00004C67 EB07               <1>         JMP     SHORT SETBAC        ; BACK
                            <1> M_12:
00004C69 808F[3D810100]06   <1>         OR      byte [DSK_STATE+eDI],DRV_DET+FMT_CAPA
                            <1>                                     ; TURN ON DETERMINED & FMT CAPA
                            <1> SETBAC:
00004C70 C3                 <1>         RETn
                            <1>
                            <1> ;--------------------------------------------------------------------------
                            <1> ; RETRY
                            <1> ;     DETERMINES WHETHER A RETRY IS NECESSARY.
                            <1> ;     IF RETRY IS REQUIRED THEN STATE INFORMATION IS UPDATED FOR RETRY.
                            <1> ;
                            <1> ; ON EXIT:  CY = 1 FOR RETRY, CY = 0 FOR NO RETRY
                            <1> ;--------------------------------------------------------------------------
                            <1> RETRY:
00004C71 803D[30810100]00   <1>         CMP     byte [DSKETTE_STATUS],0    ; GET STATUS OF OPERATION
00004C78 7445               <1>         JZ      short NO_RETRY             ; SUCCESSFUL OPERATION
00004C7A 803D[30810100]80   <1>         CMP     byte [DSKETTE_STATUS],TIME_OUT ; IF TIME OUT NO RETRY
00004C81 743C               <1>         JZ      short NO_RETRY
00004C83 8AA7[3D810100]     <1>         MOV     AH,[DSK_STATE+eDI]  ; GET MEDIA STATE OF DRIVE
00004C89 F6C410             <1>         TEST    AH,MED_DET          ; ESTABLISHED/DETERMINED ?
00004C8C 7531               <1>         JNZ     short NO_RETRY             ; IF ESTABLISHED STATE THEN TRUE ERROR
00004C8E 80E4C0             <1>         AND     AH,RATE_MSK         ; ISOLATE RATE
00004C91 8A2D[38810100]     <1>         MOV     CH,[LASTRATE]       ; GET START OPERATION STATE
00004C97 C0C504             <1>         ROL     CH,4                ; TO CORRESPONDING BITS
00004C9A 80E5C0             <1>         AND     CH,RATE_MSK         ; ISOLATE RATE BITS
00004C9D 38E5               <1>         CMP     CH,AH               ; ALL RATES TRIED
00004C9F 741E               <1>         JE      short NO_RETRY             ; IF YES, THEN TRUE ERROR
                            <1>
                            <1> ;     SETUP STATE INDICATOR FOR RETRY ATTEMPT TO NEXT RATE
                            <1> ;      00000000B (500) -> 10000000B    (250)
                            <1> ;      10000000B (250) -> 01000000B    (300)
                            <1> ;      01000000B (300) -> 00000000B    (500)
                            <1>
00004CA1 80FC01             <1>         CMP     AH,RATE_500+1       ; SET CY FOR RATE 500
00004CA4 D0DC               <1>         RCR     AH,1                ; TO NEXT STATE
00004CA6 80E4C0             <1>         AND     AH,RATE_MSK         ; KEEP ONLY RATE BITS
00004CA9 80A7[3D810100]1F   <1>         AND     byte [DSK_STATE+eDI], ~(RATE_MSK+DBL_STEP)
                            <1>                                     ; RATE, DBL STEP OFF
00004CB0 08A7[3D810100]     <1>         OR      [DSK_STATE+eDI],AH  ; TURN ON NEW RATE
00004CB6 C605[30810100]00   <1>         MOV     byte [DSKETTE_STATUS],0    ; RESET STATUS FOR RETRY
00004CBD F9                 <1>         STC                         ; SET CARRY FOR RETRY
00004CBE C3                 <1>         RETn                        ; RETRY RETURN
                            <1>
                            <1> NO_RETRY:
00004CBF F8                 <1>         CLC                         ; CLEAR CARRY NO RETRY
00004CC0 C3                 <1>         RETn                        ; NO RETRY RETURN
                            <1>
                            <1> ;--------------------------------------------------------------------------
                            <1> ; NUM_TRANS
                            <1> ;     THIS ROUTINE CALCULATES THE NUMBER OF SECTORS THAT WERE
                            <1> ;     ACTUALLY TRANSFERRED TO/FROM THE DISKETTE.
                            <1> ;
                            <1> ; ON ENTRY:  [BP+1] = TRACK
                            <1> ;            SI-HI  = HEAD
                            <1> ;            [BP]   = START SECTOR
                            <1> ;
                            <1> ; ON EXIT:   AL = NUMBER ACTUALLY TRANSFERRED
                            <1> ;--------------------------------------------------------------------------
                            <1> NUM_TRANS:
00004CC1 30C0               <1>         XOR     AL,AL               ; CLEAR FOR ERROR
00004CC3 803D[30810100]00   <1>         CMP     byte [DSKETTE_STATUS],0    ; CHECK FOR ERROR
00004CCA 752C               <1>         JNZ     NT_OUT              ; IF ERROR 0 TRANSFERRED
00004CCC B204               <1>         MOV     DL,4                ; SECTORS/TRACK OFFSET TO DL
00004CCE E8F4000000         <1>         CALL    GET_PARM            ; AH = SECTORS/TRACK
00004CD3 8A1D[36810100]     <1>         MOV     BL, [NEC_STATUS+5]  ; GET ENDING SECTOR
00004CD9 6689F1             <1>         MOV     CX,SI               ; CH = HEAD # STARTED
00004CDC 3A2D[35810100]     <1>         CMP     CH, [NEC_STATUS+4]  ; GET HEAD ENDED UP ON
```

```
5147 00004CE2 750D                <1>       JNZ    DIF_HD              ; IF ON SAME HEAD, THEN NO ADJUST
5148 00004CE4 8A2D[34810100]      <1>       MOV    CH, [NEC_STATUS+3]  ; GET TRACK ENDED UP ON
5149 00004CEA 3A6D01              <1>       CMP    CH,[eBP+1]          ; IS IT ASKED FOR TRACK
5150 00004CED 7404                <1>       JZ     short SAME_TRK            ; IF SAME TRACK NO INCREASE
5151 00004CEF 00E3                <1>       ADD    BL,AH               ; ADD SECTORS/TRACK
5152                              <1> DIF_HD:
5153 00004CF1 00E3                <1>       ADD    BL,AH               ; ADD SECTORS/TRACK
5154                              <1> SAME_TRK:
5155 00004CF3 2A5D00              <1>       SUB    BL,[eBP]            ; SUBTRACT START FROM END
5156 00004CF6 88D8                <1>       MOV    AL,BL               ; TO AL
5157                              <1> NT_OUT:
5158 00004CF8 C3                  <1>       RETn
5159                              <1>
5160                              <1> ;--------------------------------------------------------------------------
5161                              <1> ; SETUP_END
5162                              <1> ;     RESTORES @MOTOR_COUNT TO PARAMETER PROVIDED IN TABLE
5163                              <1> ;     AND LOADS @DSKETTE_STATUS TO AH, AND SETS CY.
5164                              <1> ;
5165                              <1> ; ON EXIT:
5166                              <1> ;     AH, @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
5167                              <1> ;--------------------------------------------------------------------------
5168                              <1> SETUP_END:
5169 00004CF9 B202                <1>       MOV    DL,2                ; GET THE MOTOR WAIT PARAMETER
5170                              <1>       ;PUSH  AX                  ; SAVE NUMBER TRANSFERRED
5171                              <1>       ; 11/04/2021
5172 00004CFB 50                  <1>       push   eax
5173 00004CFC E8C6000000          <1>       CALL   GET_PARM
5174 00004D01 8825[2F810100]      <1>       MOV    [MOTOR_COUNT],AH    ; STORE UPON RETURN
5175                              <1>       ;POP   AX                  ; RESTORE NUMBER TRANSFERRED
5176                              <1>       ; 11/04/2021
5177 00004D07 58                  <1>       pop    eax
5178 00004D08 8A25[30810100]      <1>       MOV    AH, [DSKETTE_STATUS]      ; GET STATUS OF OPERATION
5179 00004D0E 08E4                <1>       OR     AH,AH               ; CHECK FOR ERROR
5180 00004D10 7402                <1>       JZ     short NUN_ERR       ; NO ERROR
5181 00004D12 30C0                <1>       XOR    AL,AL               ; CLEAR NUMBER RETURNED
5182                              <1> NUN_ERR:
5183 00004D14 80FC01              <1>       CMP    AH,1                ; SET THE CARRY FLAG TO INDICATE
5184 00004D17 F5                  <1>       CMC                        ; SUCCESS OR FAILURE
5185 00004D18 C3                  <1>       RETn
5186                              <1>
5187                              <1> ;--------------------------------------------------------------------------
5188                              <1> ; SETUP_DBL
5189                              <1> ;     CHECK DOUBLE STEP.
5190                              <1> ;
5191                              <1> ; ON ENTRY : DI = DRIVE
5192                              <1> ;
5193                              <1> ; ON EXIT :  CY = 1 MEANS ERROR
5194                              <1> ;--------------------------------------------------------------------------
5195                              <1> SETUP_DBL:
5196 00004D19 8AA7[3D810100]      <1>       MOV    AH, [DSK_STATE+eDI] ; ACCESS STATE
5197 00004D1F F6C410              <1>       TEST   AH,MED_DET          ; ESTABLISHED STATE ?
5198 00004D22 757A                <1>       JNZ    short NO_DBL              ; IF ESTABLISHED THEN DOUBLE DONE
5199                              <1>
5200                              <1> ;-----      CHECK FOR TRACK 0 TO SPEED UP ACKNOWLEDGE OF UNFORMATTED DISKETTE
5201                              <1>
5202 00004D24 C605[2D810100]00    <1>       MOV    byte [SEEK_STATUS],0      ; SET RECALIBRATE REQUIRED ON ALL DRIVES
5203 00004D2B E8DC000000          <1>       CALL   MOTOR_ON            ; ENSURE MOTOR STAY ON
5204 00004D30 B500                <1>       MOV    CH,0                ; LOAD TRACK 0
5205 00004D32 E8D0010000          <1>       CALL   SEEK                ; SEEK TO TRACK 0
5206 00004D37 E864000000          <1>       CALL   READ_ID                   ; READ ID FUNCTION
5207 00004D3C 7245                <1>       JC     short SD_ERR        ; IF ERROR NO TRACK 0
5208                              <1>
5209                              <1> ;-----      INITIALIZE START AND MAX TRACKS (TIMES 2 FOR BOTH HEADS)
5210                              <1>
5211 00004D3E 66B95004            <1>       MOV    CX,0450H            ; START, MAX TRACKS
5212 00004D42 F687[3D810100]01    <1>       TEST   byte [DSK_STATE+eDI],TRK_CAPA ; TEST FOR 80 TRACK CAPABILITY
5213 00004D49 7402                <1>       JZ     short CNT_OK        ; IF NOT COUNT IS SETUP
5214 00004D4B B1A0                <1>       MOV    CL,0A0H                  ; MAXIMUM TRACK 1.2 MB
5215                              <1>
5216                              <1> ;     ATTEMPT READ ID OF ALL TRACKS, ALL HEADS UNTIL SUCCESS; UPON SUCCESS,
5217                              <1> ;     MUST SEE IF ASKED FOR TRACK IN SINGLE STEP MODE = TRACK ID READ; IF NOT
5218                              <1> ;     THEN SET DOUBLE STEP ON.
5219                              <1>
5220                              <1> CNT_OK:
5221                              <1>       ; 11/04/2021 (32 bit push/pop)
5222 00004D4D C605[2F810100]FF    <1>         MOV     byte [MOTOR_COUNT], 0FFH ; ENSURE MOTOR STAYS ON FOR OPERATION
5223 00004D54 51                  <1>       PUSH   eCX                 ; SAVE TRACK, COUNT
5224 00004D55 C605[30810100]00    <1>       MOV    byte [DSKETTE_STATUS],0   ; CLEAR STATUS, EXPECT ERRORS
5225 00004D5C 6631C0              <1>       XOR    AX,AX               ; CLEAR AX
5226 00004D5F D0ED                <1>       SHR    CH,1                ; HALVE TRACK, CY = HEAD
5227 00004D61 C0D003              <1>       RCL    AL,3                ; AX = HEAD IN CORRECT BIT
5228 00004D64 50                  <1>       PUSH   eAX                 ; SAVE HEAD
5229 00004D65 E89D010000          <1>       CALL   SEEK                ; SEEK TO TRACK
5230 00004D6A 58                  <1>       POP    eAX                 ; RESTORE HEAD
5231 00004D6B 6609C7              <1>       OR     DI,AX               ; DI = HEAD OR'ED DRIVE
5232 00004D6E E82D000000          <1>       CALL   READ_ID                   ; READ ID HEAD 0
5233 00004D73 9C                  <1>       PUSHF                      ; SAVE RETURN FROM READ_ID
5234 00004D74 6681E7FB00          <1>       AND    DI,11111011B        ; TURN OFF HEAD 1 BIT
5235 00004D79 9D                  <1>       POPF                       ; RESTORE ERROR RETURN
5236 00004D7A 59                  <1>       POP    eCX                 ; RESTORE COUNT
5237 00004D7B 7308                <1>       JNC    short DO_CHK        ; IF OK, ASKED = RETURNED TRACK ?
5238 00004D7D FEC5                <1>       INC    CH                  ; INC FOR NEXT TRACK
5239 00004D7F 38CD                <1>       CMP    CH,CL               ; REACHED MAXIMUM YET
5240 00004D81 75CA                <1>       JNZ    short CNT_OK        ; CONTINUE TILL ALL TRIED
5241                              <1>
5242                              <1> ;-----      FALL THRU, READ ID FAILED FOR ALL TRACKS
5243                              <1>
5244                              <1> SD_ERR:
5245 00004D83 F9                  <1>       STC                        ; SET CARRY FOR ERROR
5246 00004D84 C3                  <1>       RETn                       ; SETUP_DBL ERROR EXIT
5247                              <1>
5248                              <1> DO_CHK:
5249 00004D85 8A0D[34810100]      <1>       MOV    CL, [NEC_STATUS+3]  ; LOAD RETURNED TRACK
5250 00004D8B 888F[41810100]      <1>       MOV    [DSK_TRK+eDI], CL   ; STORE TRACK NUMBER
5251 00004D91 D0ED                <1>       SHR    CH,1                ; HALVE TRACK
```

```
5252 00004D93 38CD                <1>       CMP    CH,CL               ; IS IT THE SAME AS ASKED FOR TRACK
5253 00004D95 7407                <1>       JZ     short NO_DBL        ; IF SAME THEN NO DOUBLE STEP
5254 00004D97 808F[3D810100]20    <1>       OR     byte [DSK_STATE+eDI],DBL_STEP ; TURN ON DOUBLE STEP REQUIRED
5255                              <1> NO_DBL:
5256 00004D9E F8                  <1>       CLC                        ; CLEAR ERROR FLAG
5257 00004D9F C3                  <1>       RETn
5258                              <1>
5259                              <1> ;--------------------------------------------------------------------------
5260                              <1> ; READ_ID
5261                              <1> ;     READ ID FUNCTION.
5262                              <1> ;
5263                              <1> ; ON ENTRY:  DI : BIT 2 = HEAD; BITS 1,0 = DRIVE
5264                              <1> ;
5265                              <1> ; ON EXIT:   DI : BIT 2 IS RESET, BITS 1,0 = DRIVE
5266                              <1> ;            @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
5267                              <1> ;--------------------------------------------------------------------------
5268                              <1> READ_ID:
5269 00004DA0 B8[BD4D0000]        <1>       MOV    eAX, ER_3           ; MOVE NEC OUTPUT ERROR ADDRESS
5270 00004DA5 50                  <1>       PUSH   eAX
5271 00004DA6 B44A                <1>       MOV    AH,4AH              ; READ ID COMMAND
5272 00004DA8 E820010000          <1>       CALL   NEC_OUTPUT          ; TO CONTROLLER
5273 00004DAD 6689F8              <1>       MOV    AX,DI               ; DRIVE # TO AH, HEAD 0
5274 00004DB0 88C4                <1>       MOV    AH,AL
5275 00004DB2 E816010000          <1>       CALL   NEC_OUTPUT          ; TO CONTROLLER
5276 00004DB7 E811FEFFFF          <1>       CALL   NEC_TERM            ; WAIT FOR OPERATION, GET STATUS
5277 00004DBC 58                  <1>       POP    eAX                 ; THROW AWAY ERROR ADDRESS
5278                              <1> ER_3:
5279 00004DBD C3                  <1>       RETn
5280                              <1>
5281                              <1> ;--------------------------------------------------------------------------
5282                              <1> ; CMOS_TYPE
5283                              <1> ;     RETURNS DISKETTE TYPE FROM CMOS
5284                              <1> ;
5285                              <1> ; ON ENTRY:  DI = DRIVE #
5286                              <1> ;
5287                              <1> ; ON EXIT:   AL = TYPE; CY REFLECTS STATUS
5288                              <1> ;--------------------------------------------------------------------------
5289                              <1>
5290                              <1> CMOS_TYPE: ; 11/12/2014
5291 00004DBE 8A87[EE680000]      <1>       mov    al, [eDI+fd0_type]
5292 00004DC4 20C0                <1>       and    al, al ; 18/12/2014
5293 00004DC6 C3                  <1>       retn
5294                              <1>
5295                              <1> ;CMOS_TYPE:
5296                              <1> ;     MOV    AL, CMOS_DIAG       ; CMOS DIAGNOSTIC STATUS BYTE ADDRESS
5297                              <1> ;     CALL   CMOS_READ           ; GET CMOS STATUS
5298                              <1> ;     TEST   AL,BAD_BAT+BAD_CKSUM      ; BATTERY GOOD AND CHECKSUM VALID
5299                              <1> ;     STC                        ; SET CY = 1 INDICATING ERROR FOR RETURN
5300                              <1> ;     JNZ    short BAD_CM        ; ERROR IF EITHER BIT ON
5301                              <1> ;     MOV    AL,CMOS_DISKETTE    ; ADDRESS OF DISKETTE BYTE IN CMOS
5302                              <1> ;     CALL   CMOS_READ           ; GET DISKETTE BYTE
5303                              <1> ;     OR     DI,DI               ; SEE WHICH DRIVE IN QUESTION
5304                              <1> ;     JNZ    short TB            ; IF DRIVE 1, DATA IN LOW NIBBLE
5305                              <1> ;     ROR    AL,4                ; EXCHANGE NIBBLES IF SECOND DRIVE
5306                              <1> ;TB:
5307                              <1> ;     AND    AL,0FH              ; KEEP ONLY DRIVE DATA, RESET CY, 0
5308                              <1> ;BAD_CM:
5309                              <1> ;     RETn                       ; CY, STATUS OF READ
5310                              <1>
5311                              <1> ;--------------------------------------------------------------------------
5312                              <1> ; GET_PARM
5313                              <1> ;     THIS ROUTINE FETCHES THE INDEXED POINTER FROM THE DISK_BASE
5314                              <1> ;     BLOCK POINTED TO BY THE DATA VARIABLE @DISK_POINTER. A BYTE FROM
5315                              <1> ;     THAT TABLE IS THEN MOVED INTO AH, THE INDEX OF THAT BYTE BEING
5316                              <1> ;     THE PARAMETER IN DL.
5317                              <1> ;
5318                              <1> ; ON ENTRY:  DL = INDEX OF BYTE TO BE FETCHED
5319                              <1> ;
5320                              <1> ; ON EXIT:   AH = THAT BYTE FROM BLOCK
5321                              <1> ;            AL,DH DESTROYED
5322                              <1> ;--------------------------------------------------------------------------
5323                              <1> GET_PARM:
5324                              <1>       ;PUSH  DS
5325 00004DC7 56                  <1>       PUSH   eSI
5326                              <1>       ;SUB   AX,AX               ; DS = 0, BIOS DATA AREA
5327                              <1>       ;MOV   DS,AX
5328                              <1>       ;;mov  ax, cs
5329                              <1>       ;;mov  ds, ax
5330                              <1>       ; 08/02/2015 (protected mode modifications, bx -> ebx)
5331 00004DC8 87D3                <1>       XCHG   eDX,eBX                   ; BL = INDEX
5332                              <1>       ;SUB   BH,BH               ; BX = INDEX
5333 00004DCA 81E3FF000000        <1>       and    ebx, 0FFh
5334                              <1>       ;LDS   SI, [DISK_POINTER]  ; POINT TO BLOCK
5335                              <1>       ;
5336                              <1>       ; 17/12/2014
5337 00004DD0 66A1[DD680000]      <1>       mov    ax, [cfd] ; current (AL) and previous fd (AH)
5338 00004DD6 38E0                <1>       cmp    al, ah
5339 00004DD8 7425                <1>       je     short gpndc
5340 00004DDA A2[DE680000]        <1>       mov    [pfd], al ; current drive -> previous drive
5341 00004DDF 53                  <1>       push   ebx ; 08/02/2015
5342 00004DE0 88C3                <1>       mov    bl, al
5343                              <1>       ; 11/12/2014
5344 00004DE2 8A83[EE680000]      <1>       mov    al, [eBX+fd0_type]  ; Drive type (0,1,2,3,4)
5345                              <1>       ; 18/12/2014
5346 00004DE8 20C0                <1>       and    al, al
5347 00004DEA 7507                <1>       jnz    short gpdtc
5348 00004DEC BB[C7680000]        <1>       mov    ebx, MD_TBL6        ; 1.44 MB param. tbl. (default)
5349 00004DF1 EB05                <1>       jmp    short gpdpu
5350                              <1> gpdtc:
5351 00004DF3 E835F9FFFF          <1>       call   DR_TYPE_CHECK
5352                              <1>       ; cf = 1 -> eBX points to 1.44MB fd parameter table (default)
5353                              <1> gpdpu:
5354 00004DF8 891D[64680000]      <1>       mov    [DISK_POINTER], ebx
5355 00004DFE 5B                  <1>       pop    ebx
5356                              <1> gpndc:
```

```
5357 00004DFF 8B35[64680000]      <1>       mov    esi, [DISK_POINTER] ; 08/02/2015, si -> esi
5358 00004E05 8A241E              <1>       MOV    AH, [eSI+eBX]       ; GET THE WORD
5359 00004E08 87D3                <1>       XCHG   eDX,eBX                     ; RESTORE BX
5360 00004E0A 5E                  <1>       POP    eSI
5361                              <1>       ;POP   DS
5362 00004E0B C3                  <1>       RETn
5363                              <1>
5364                              <1> ;--------------------------------------------------------------------------
5365                              <1> ; MOTOR_ON
5366                              <1> ;     TURN MOTOR ON AND WAIT FOR MOTOR START UP TIME. THE @MOTOR_COUNT
5367                              <1> ;     IS REPLACED WITH A SUFFICIENTLY HIGH NUMBER (0FFH) TO ENSURE
5368                              <1> ;     THAT THE MOTOR DOES NOT GO OFF DURING THE OPERATION. IF THE
5369                              <1> ;     MOTOR NEEDED TO BE TURNED ON, THE MULTI-TASKING HOOK FUNCTION
5370                              <1> ;     (AX=90FDH, INT 15) IS CALLED TELLING THE OPERATING SYSTEM
5371                              <1> ;     THAT THE BIOS IS ABOUT TO WAIT FOR MOTOR START UP. IF THIS
5372                              <1> ;     FUNCTION RETURNS WITH CY = 1, IT MEANS THAT THE MINIMUM WAIT
5373                              <1> ;     HAS BEEN COMPLETED. AT THIS POINT A CHECK IS MADE TO ENSURE
5374                              <1> ;     THAT THE MOTOR WASN'T TURNED OFF BY THE TIMER. IF THE HOOK DID
5375                              <1> ;     NOT WAIT, THE WAIT FUNCTION (AH=086H) IS CALLED TO WAIT THE
5376                              <1> ;     PRESCRIBED AMOUNT OF TIME. IF THE CARRY FLAG IS SET ON RETURN,
5377                              <1> ;     IT MEANS THAT THE FUNCTION IS IN USE AND DID NOT PERFORM THE
5378                              <1> ;     WAIT. A TIMER 1 WAIT LOOP WILL THEN DO THE WAIT.
5379                              <1> ;
5380                              <1> ; ON ENTRY:  DI = DRIVE #
5381                              <1> ; ON EXIT:   AX,CX,DX DESTROYED
5382                              <1> ;--------------------------------------------------------------------------
5383                              <1> MOTOR_ON:
5384 00004E0C 53                  <1>       PUSH   eBX                 ; SAVE REG.
5385 00004E0D E82A000000          <1>       CALL   TURN_ON             ; TURN ON MOTOR
5386 00004E12 7226                <1>       JC     short MOT_IS_ON     ; IF CY=1 NO WAIT
5387 00004E14 E8B7F9FFFF          <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
5388 00004E19 E881F9FFFF          <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH,
5389                              <1>       ;CALL  TURN_ON             ; CHECK AGAIN IF MOTOR ON
5390                              <1>       ;JC    MOT_IS_ON           ; IF NO WAIT MEANS IT IS ON
5391                              <1> M_WAIT:
5392 00004E1E B20A                <1>       MOV    DL,10               ; GET THE MOTOR WAIT PARAMETER
5393 00004E20 E8A2FFFFFF          <1>       CALL   GET_PARM
5394                              <1>       ;MOV   AL,AH               ; AL = MOTOR WAIT PARAMETER
5395                              <1>       ;XOR   AH,AH               ; AX = MOTOR WAIT PARAMETER
5396                              <1>       ;CMP   AL,8                ; SEE IF AT LEAST A SECOND IS SPECIFIED
5397 00004E25 80FC08              <1>       cmp    ah, 8
5398                              <1>       ;JAE   short GP2           ; IF YES, CONTINUE
5399 00004E28 7702                <1>       ja     short J13
5400                              <1>       ;MOV   AL,8                ; ONE SECOND WAIT FOR MOTOR START UP
5401 00004E2A B408                <1>       mov    ah, 8
5402                              <1>
5403                              <1> ;----- AS CONTAINS NUMBER OF 1/8 SECONDS (125000 MICROSECONDS) TO WAIT
5404                              <1> GP2:
5405                              <1> ;----- FOLLOWING LOOPS REQUIRED WHEN RTC WAIT FUNCTION IS ALREADY IN USE
5406                              <1> J13:                           ; WAIT FOR 1/8 SECOND PER (AL)
5407                              <1>       ;MOV   eCX,8286          ; COUNT FOR 1/8 SECOND AT 15.085737 US
5408                              <1>       ; 11/04/2021
5409 00004E2C B947100000          <1>       mov    ecx, 4167 ; count of 30 micro seconds * (1/8)
5410 00004E31 E872D5FFFF          <1>       CALL   WAITF              ; GO TO FIXED WAIT ROUTINE
5411                              <1>       ;DEC   AL                 ; DECREMENT TIME VALUE
5412 00004E36 FECC                <1>       dec    ah
5413 00004E38 75F2                <1>       JNZ    short J13          ; ARE WE DONE YET
5414                              <1> MOT_IS_ON:
5415 00004E3A 5B                  <1>       POP    eBX                ; RESTORE REG.
5416 00004E3B C3                  <1>       RETn
5417                              <1>
5418                              <1> ;--------------------------------------------------------------------------
5419                              <1> ; TURN_ON
5420                              <1> ;     TURN MOTOR ON AND RETURN WAIT STATE.
5421                              <1> ;
5422                              <1> ; ON ENTRY:  DI = DRIVE #
5423                              <1> ;
5424                              <1> ; ON EXIT:   CY = 0 MEANS WAIT REQUIRED
5425                              <1> ;            CY = 1 MEANS NO WAIT REQUIRED
5426                              <1> ;            AX,BX,CX,DX DESTROYED
5427                              <1> ;--------------------------------------------------------------------------
5428                              <1> TURN_ON:
5429 00004E3C 89FB                <1>       MOV    eBX,eDI            ; BX = DRIVE #
5430 00004E3E 88D9                <1>       MOV    CL,BL              ; CL = DRIVE #
5431 00004E40 C0C304              <1>       ROL    BL,4               ; BL = DRIVE SELECT
5432 00004E43 FA                  <1>       CLI                       ; NO INTERRUPTS WHILE DETERMINING STATUS
5433 00004E44 C605[2F810100]FF    <1>       MOV    byte [MOTOR_COUNT],0FFH   ; ENSURE MOTOR STAYS ON FOR OPERATION
5434 00004E4B A0[2E810100]        <1>       MOV    AL, [MOTOR_STATUS] ; GET DIGITAL OUTPUT REGISTER REFLECTION
5435 00004E50 2430                <1>       AND    AL,00110000B       ; KEEP ONLY DRIVE SELECT BITS
5436 00004E52 B401                <1>       MOV    AH,1               ; MASK FOR DETERMINING MOTOR BIT
5437 00004E54 D2E4                <1>       SHL    AH,CL              ; AH = MOTOR ON, A=00000001, B=00000010
5438                              <1>
5439                              <1> ;  AL = DRIVE SELECT FROM @MOTOR_STATUS
5440                              <1> ;  BL = DRIVE SELECT DESIRED
5441                              <1> ;  AH = MOTOR ON MASK DESIRED
5442                              <1>
5443 00004E56 38D8                <1>       CMP    AL,BL              ; REQUESTED DRIVE ALREADY SELECTED ?
5444 00004E58 7508                <1>       JNZ    short TURN_IT_ON   ; IF NOT SELECTED JUMP
5445 00004E5A 8425[2E810100]      <1>       TEST   AH, [MOTOR_STATUS] ; TEST MOTOR ON BIT
5446 00004E60 7535                <1>       JNZ    short NO_MOT_WAIT  ; JUMP IF MOTOR ON AND SELECTED
5447                              <1>
5448                              <1> TURN_IT_ON:
5449 00004E62 08DC                <1>       OR     AH,BL              ; AH = DRIVE SELECT AND MOTOR ON
5450 00004E64 8A3D[2E810100]      <1>       MOV    BH,[MOTOR_STATUS]  ; SAVE COPY OF @MOTOR_STATUS BEFORE
5451 00004E6A 80E70F              <1>       AND    BH,00001111B       ; KEEP ONLY MOTOR BITS
5452 00004E6D 8025[2E810100]CF    <1>       AND    byte [MOTOR_STATUS],11001111B ; CLEAR OUT DRIVE SELECT
5453 00004E74 0825[2E810100]      <1>       OR     [MOTOR_STATUS],AH  ; OR IN DRIVE SELECTED AND MOTOR ON
5454 00004E7A A0[2E810100]        <1>       MOV    AL,[MOTOR_STATUS]  ; GET DIGITAL OUTPUT REGISTER REFLECTION
5455 00004E7F 88C3                <1>       MOV    BL,AL              ; BL=@MOTOR_STATUS AFTER, BH=BEFORE
5456 00004E81 80E30F              <1>       AND    BL,00001111B       ; KEEP ONLY MOTOR BITS
5457 00004E84 FB                  <1>       STI                       ; ENABLE INTERRUPTS AGAIN
5458 00004E85 243F                <1>       AND    AL,00111111B       ; STRIP AWAY UNWANTED BITS
5459 00004E87 C0C004              <1>       ROL    AL,4               ; PUT BITS IN DESIRED POSITIONS
5460 00004E8A 0C0C                <1>       OR     AL,00001100B       ; NO RESET, ENABLE DMA/INTERRUPT
5461 00004E8C 66BAF203            <1>       MOV    DX,03F2H           ; SELECT DRIVE AND TURN ON MOTOR
```

```
5462 00004E90 EE                <1>       OUT    DX,AL
5463 00004E91 38FB              <1>       CMP    BL,BH               ; NEW MOTOR TURNED ON ?
5464                            <1>       ;JZ    short NO_MOT_WAIT   ; NO WAIT REQUIRED IF JUST SELECT
5465 00004E93 7403              <1>       je     short no_mot_w1 ; 27/02/2015
5466 00004E95 F8                <1>       CLC                        ; (re)SET CARRY MEANING WAIT
5467 00004E96 C3                <1>       RETn
5468                            <1>
5469                            <1> NO_MOT_WAIT:
5470 00004E97 FB                <1>       sti
5471                            <1> no_mot_w1: ; 27/02/2015
5472 00004E98 F9                <1>       STC                        ; SET NO WAIT REQUIRED
5473                            <1>       ;STI                       ; INTERRUPTS BACK ON
5474 00004E99 C3                <1>       RETn
5475                            <1>
5476                            <1> ;--------------------------------------------------------------------------
5477                            <1> ; HD_WAIT
5478                            <1> ;     WAIT FOR HEAD SETTLE TIME.
5479                            <1> ;
5480                            <1> ; ON ENTRY:  DI = DRIVE #
5481                            <1> ;
5482                            <1> ; ON EXIT:   AX,BX,CX,DX DESTROYED
5483                            <1> ;--------------------------------------------------------------------------
5484                            <1> HD_WAIT:
5485 00004E9A B209              <1>       MOV    DL,9                ; GET HEAD SETTLE PARAMETER
5486 00004E9C E826FFFFFF        <1>       CALL   GET_PARM
5487 00004EA1 08E4              <1>       or     ah, ah ; 17/12/2014
5488 00004EA3 7519              <1>       jnz    short DO_WAT
5489 00004EA5 F605[2E810100]80  <1>       TEST  byte [MOTOR_STATUS],10000000B ; SEE IF A WRITE OPERATION
5490                            <1>       ;JZ    short ISNT_WRITE    ; IF NOT, DO NOT ENFORCE ANY VALUES
5491                            <1>       ;OR    AH,AH               ; CHECK FOR ANY WAIT?
5492                            <1>       ;JNZ   short DO_WAT        ; IF THERE DO NOT ENFORCE
5493 00004EAC 741E              <1>       jz     short HW_DONE
5494 00004EAE B40F              <1>       MOV    AH,HD12_SETTLE           ; LOAD 1.2M HEAD SETTLE MINIMUM
5495 00004EB0 8A87[3D810100]    <1>       MOV    AL,[DSK_STATE+eDI]  ; LOAD STATE
5496 00004EB6 24C0              <1>       AND    AL,RATE_MSK         ; KEEP ONLY RATE
5497 00004EB8 3C80              <1>       CMP    AL,RATE_250         ; 1.2 M DRIVE ?
5498 00004EBA 7502              <1>       JNZ    short DO_WAT        ; DEFAULT HEAD SETTLE LOADED
5499                            <1> ;GP3:
5500 00004EBC B414              <1>       MOV    AH,HD320_SETTLE           ; USE 320/360 HEAD SETTLE
5501                            <1> ;     JMP    SHORT DO_WAT
5502                            <1>
5503                            <1> ;ISNT_WRITE:
5504                            <1> ;     OR     AH,AH               ; CHECK FOR NO WAIT
5505                            <1> ;     JZ     short HW_DONE       ; IF NOT WRITE AND 0 ITS OK
5506                            <1>
5507                            <1> ;----- AH CONTAINS NUMBER OF MILLISECONDS TO WAIT
5508                            <1> DO_WAT:
5509                            <1> ;     MOV    AL,AH               ; AL = # MILLISECONDS
5510                            <1> ;     ;XOR   AH,AH               ; AX = # MILLISECONDS
5511                            <1> J29:                             ;      1 MILLISECOND LOOP
5512                            <1>       ;;mov  cx, WAIT_FDU_HEAD_SETTLE ; 33 ; 1 ms in 30 micro units.
5513                            <1>       ;MOV   eCX,66              ; COUNT AT 15.085737 US PER COUNT
5514                            <1>       ; 11/04/2021
5515 00004EBE B921000000        <1>       mov    ecx, WAIT_FDU_HEAD_SETTLE ; 33
5516 00004EC3 E8E0D4FFFF        <1>       CALL   WAITF               ; DELAY FOR 1 MILLISECOND
5517                            <1>       ;DEC   AL                  ; DECREMENT THE COUNT
5518 00004EC8 FECC              <1>       dec    ah
5519 00004ECA 75F2              <1>       JNZ    short J29           ; DO AL MILLISECOND # OF TIMES
5520                            <1> HW_DONE:
5521 00004ECC C3                <1>       RETn
5522                            <1>
5523                            <1> ;--------------------------------------------------------------------------
5524                            <1> ; NEC_OUTPUT
5525                            <1> ;     THIS ROUTINE SENDS A BYTE TO THE NEC CONTROLLER AFTER TESTING
5526                            <1> ;     FOR CORRECT DIRECTION AND CONTROLLER READY THIS ROUTINE WILL
5527                            <1> ;     TIME OUT IF THE BYTE IS NOT ACCEPTED WITHIN A REASONABLE AMOUNT
5528                            <1> ;     OF TIME, SETTING THE DISKETTE STATUS ON COMPLETION.
5529                            <1> ;
5530                            <1> ; ON ENTRY:  AH = BYTE TO BE OUTPUT
5531                            <1> ;
5532                            <1> ; ON EXIT:   CY = 0  SUCCESS
5533                            <1> ;            CY = 1  FAILURE -- DISKETTE STATUS UPDATED
5534                            <1> ;                    IF A FAILURE HAS OCCURRED, THE RETURN IS MADE ONE LEVEL
5535                            <1> ;                    HIGHER THAN THE CALLER OF NEC OUTPUT. THIS REMOVES THE
5536                            <1> ;                    REQUIREMENT OF TESTING AFTER EVERY CALL OF NEC_OUTPUT.
5537                            <1> ;            AX,CX,DX DESTROYED
5538                            <1> ;--------------------------------------------------------------------------
5539                            <1>
5540                            <1> ; 09/12/2014 [Erdogan Tan]
5541                            <1> ;     (from 'PS2 Hardware Interface Tech. Ref. May 88', Page 09-05.)
5542                            <1> ; Diskette Drive Controller Status Register (3F4h)
5543                            <1> ;     This read only register facilitates the transfer of data between
5544                            <1> ;     the system microprocessor and the controller.
5545                            <1> ; Bit 7 - When set to 1, the Data register is ready to transfer data
5546                            <1> ;       with the system micrprocessor.
5547                            <1> ; Bit 6 - The direction of data transfer. If this bit is set to 0,
5548                            <1> ;       the transfer is to the controller.
5549                            <1> ; Bit 5 - When this bit is set to 1, the controller is in the non-DMA mode.
5550                            <1> ; Bit 4 - When this bit is set to 1, a Read or Write command is being executed.
5551                            <1> ; Bit 3 - Reserved.
5552                            <1> ; Bit 2 - Reserved.
5553                            <1> ; Bit 1 - When this bit is set to 1, dskette drive 1 is in the seek mode.
5554                            <1> ; Bit 0 - When this bit is set to 1, dskette drive 1 is in the seek mode.
5555                            <1>
5556                            <1> ; Data Register (3F5h)
5557                            <1> ; This read/write register passes data, commands and parameters, and provides
5558                            <1> ; diskette status information.
5559                            <1>
5560                            <1> NEC_OUTPUT:
5561                            <1>       ;PUSH  BX                  ; SAVE REG.
5562 00004ECD 66BAF403          <1>       MOV    DX,03F4H            ; STATUS PORT
5563                            <1>       ;MOV   BL,2                ; HIGH ORDER COUNTER
5564                            <1>       ;XOR   CX,CX               ; COUNT FOR TIME OUT
5565                            <1>       ; 16/12/2014
5566                            <1>       ; waiting for (max.) 0.5 seconds
```

```
                        <1>         ;;mov     byte [wait_count], 0 ;; 27/02/2015
                        <1>       ;
                        <1>       ; 17/12/2014
                        <1>       ; Modified from AWARD BIOS 1999 - ADISK.ASM - SEND_COMMAND
                        <1>       ;
                        <1>       ;WAIT_FOR_PORT:     Waits for a bit at a port pointed to by DX to
                        <1>       ;            go on.
                        <1>       ;INPUT:
                        <1>       ;      AH=Mask for isolation bits.
                        <1>       ;      AL=pattern to look for.
                        <1>       ;      DX=Port to test for
                        <1>       ;      BH:CX=Number of memory refresh periods to delay.
                        <1>       ;           (normally 30 microseconds per period.)
                        <1>       ;
                        <1>       ;WFP_SHORT:
                        <1>       ;      Wait for port if refresh cycle is short (15-80 Us range).
                        <1>       ;
                        <1>
                        <1> ;     mov    bl, WAIT_FDU_SEND_HI+1     ; 0+1
                        <1> ;     mov    cx, WAIT_FDU_SEND_LO       ; 16667
00004ED1 B91B410000     <1>       mov    ecx, WAIT_FDU_SEND_LH   ; 16667 (27/02/2015)
                        <1> ;
                        <1> ;WFPS_OUTER_LP:
                        <1> ;     ;
                        <1> ;WFPS_CHECK_PORT:
                        <1> J23:
00004ED6 EC             <1>       IN     AL,DX               ; GET STATUS
00004ED7 24C0           <1>       AND    AL,11000000B        ; KEEP STATUS AND DIRECTION
00004ED9 3C80           <1>       CMP    AL,10000000B        ; STATUS 1 AND DIRECTION 0 ?
00004EDB 7418           <1>       JZ     short J27           ; STATUS AND DIRECTION OK
                        <1> WFPS_HI:
00004EDD E461           <1>       IN     AL, PORT_B   ;061h  ; SYS1 ; wait for hi to lo
00004EDF A810           <1>       TEST   AL,010H                    ; transition on memory
00004EE1 75FA           <1>       JNZ    SHORT WFPS_HI       ; refresh.
                        <1> WFPS_LO:
00004EE3 E461           <1>       IN     AL, PORT_B          ; SYS1
00004EE5 A810           <1>       TEST   AL,010H
00004EE7 74FA           <1>       JZ     SHORT WFPS_LO
                        <1>       ;LOOP  SHORT WFPS_CHECK_PORT
00004EE9 E2EB           <1>       loop   J23    ; 27/02/2015
                        <1> ;     ;
                        <1> ;     dec    bl
                        <1> ;     jnz    short WFPS_OUTER_LP
                        <1> ;     jmp    short WFPS_TIMEOUT  ; fail
                        <1> ;J23:
                        <1> ;     IN     AL,DX               ; GET STATUS
                        <1> ;     AND    AL,11000000B        ; KEEP STATUS AND DIRECTION
                        <1> ;     CMP    AL,10000000B        ; STATUS 1 AND DIRECTION 0 ?
                        <1> ;     JZ     short J27           ; STATUS AND DIRECTION OK
                        <1>       ;LOOP  J23                 ; CONTINUE TILL CX EXHAUSTED
                        <1>       ;DEC   BL                  ; DECREMENT COUNTER
                        <1>       ;JNZ   short J23           ; REPEAT TILL DELAY FINISHED, CX = 0
                        <1>
                        <1>       ;;27/02/2015
                        <1>       ;16/12/2014
                        <1>       ;;cmp byte [wait_count], 10 ; (10/18.2 seconds)
                        <1>       ;;jb   short J23
                        <1>
                        <1> ;WFPS_TIMEOUT:
                        <1>
                        <1> ;----- FALL THRU TO ERROR RETURN
                        <1>
00004EEB 800D[30810100]80 <1>       OR     byte [DSKETTE_STATUS],TIME_OUT
                        <1>       ;POP   BX                  ; RESTORE REG.
00004EF2 58             <1>       POP    eAX ; 08/02/2015    ; DISCARD THE RETURN ADDRESS
00004EF3 F9             <1>       STC                        ; INDICATE ERROR TO CALLER
00004EF4 C3             <1>       RETn
                        <1>
                        <1> ;----- DIRECTION AND STATUS OK; OUTPUT BYTE
                        <1>
                        <1> J27:
00004EF5 88E0           <1>       MOV    AL,AH               ; GET BYTE TO OUTPUT
00004EF7 6642           <1>       INC    DX                  ; DATA PORT = STATUS PORT + 1
00004EF9 EE             <1>       OUT    DX,AL               ; OUTPUT THE BYTE
                        <1>       ;;NEWIODELAY  ;; 27/02/2015
                        <1>       ; 27/02/2015
00004EFA 9C             <1>       PUSHF                      ; SAVE FLAGS
                        <1>       ;MOV   eCX, 3              ; 30 TO 45 MICROSECONDS WAIT FOR
                        <1>       ; 11/04/2021
00004EFB B902000000     <1>       mov    ecx, 2
00004F00 E8A3D4FFFF     <1>       CALL   WAITF               ; NEC FLAGS UPDATE CYCLE
00004F05 9D             <1>       POPF                       ; RESTORE FLAGS FOR EXIT
                        <1>       ;POP   BX                  ; RESTORE REG
00004F06 C3             <1>       RETn                       ; CY = 0 FROM TEST INSTRUCTION
                        <1>
                        <1> ;--------------------------------------------------------------------------
                        <1> ; SEEK
                        <1> ;     THIS ROUTINE WILL MOVE THE HEAD ON THE NAMED DRIVE TO THE NAMED
                        <1> ;     TRACK. IF THE DRIVE HAS NOT BEEN ACCESSED SINCE THE DRIVE
                        <1> ;     RESET COMMAND WAS ISSUED, THE DRIVE WILL BE RECALIBRATED.
                        <1> ;
                        <1> ; ON ENTRY:  DI = DRIVE #
                        <1> ;            CH = TRACK #
                        <1> ;
                        <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
                        <1> ;            AX,BX,CX DX DESTROYED
                        <1> ;--------------------------------------------------------------------------
                        <1> SEEK:
00004F07 89FB           <1>       MOV    eBX,eDI                  ; BX = DRIVE #
00004F09 B001           <1>       MOV    AL,1                ; ESTABLISH MASK FOR RECALIBRATE TEST
00004F0B 86CB           <1>       XCHG   CL,BL               ; SET DRIVE VALULE INTO CL
00004F0D D2C0           <1>       ROL    AL,CL               ; SHIFT MASK BY THE DRIVE VALUE
00004F0F 86CB           <1>       XCHG   CL,BL               ; RECOVER TRACK VALUE
00004F11 8405[2D810100] <1>       TEST   AL,[SEEK_STATUS]    ; TEST FOR RECALIBRATE REQUIRED
00004F17 7526           <1>       JNZ    short J28A          ; JUMP IF RECALIBRATE NOT REQUIRED
```

```
                                <1>
00004F19 0805[2D810100]         <1>         OR      [SEEK_STATUS],AL    ; TURN ON THE NO RECALIBRATE BIT IN FLAG
00004F1F E862000000             <1>         CALL    RECAL               ; RECALIBRATE DRIVE
00004F24 730E                   <1>         JNC     short AFT_RECAL     ; RECALIBRATE DONE
                                <1>
                                <1> ;----- ISSUE RECALIBRATE FOR 80 TRACK DISKETTES
                                <1>
00004F26 C605[30810100]00       <1>         MOV     byte [DSKETTE_STATUS],0   ; CLEAR OUT INVALID STATUS
00004F2D E854000000             <1>         CALL    RECAL               ; RECALIBRATE DRIVE
00004F32 7251                   <1>         JC      short RB            ; IF RECALIBRATE FAILS TWICE THEN ERROR
                                <1>
                                <1> AFT_RECAL:
00004F34 C687[41810100]00       <1>         MOV     byte [DSK_TRK+eDI],0 ; SAVE NEW CYLINDER AS PRESENT POSITION
00004F3B 08ED                   <1>         OR      CH,CH               ; CHECK FOR SEEK TO TRACK 0
00004F3D 743F                   <1>         JZ      short DO_WAIT       ; HEAD SETTLE, CY = 0 IF JUMP
                                <1>
                                <1> ;----- DRIVE IS IN SYNCHRONIZATION WITH CONTROLLER, SEEK TO TRACK
                                <1>
00004F3F F687[3D810100]20       <1> J28A: TEST  byte [DSK_STATE+eDI],DBL_STEP ; CHECK FOR DOUBLE STEP REQUIRED
00004F46 7402                   <1>         JZ      short _R7           ; SINGLE STEP REQUIRED BYPASS DOUBLE
00004F48 D0E5                   <1>         SHL     CH,1                ; DOUBLE NUMBER OF STEP TO TAKE
                                <1>
00004F4A 3AAF[41810100]         <1> _R7:    CMP     CH, [DSK_TRK+eDI]   ; SEE IF ALREADY AT THE DESIRED TRACK
00004F50 7433                   <1>         JE      short RB            ; IF YES, DO NOT NEED TO SEEK
                                <1>
00004F52 BA[854F0000]           <1>         MOV     eDX, NEC_ERR        ; LOAD RETURN ADDRESS
00004F57 52                     <1>         PUSH    eDX ; (*)           ; ON STACK FOR NEC OUTPUT ERROR
00004F58 88AF[41810100]         <1>         MOV     [DSK_TRK+eDI],CH    ; SAVE NEW CYLINDER AS PRESENT POSITION
00004F5E B40F                   <1>         MOV     AH,0FH              ; SEEK COMMAND TO NEC
00004F60 E868FFFFFF             <1>         CALL    NEC_OUTPUT
00004F65 89FB                   <1>         MOV     eBX,eDI             ; BX = DRIVE #
00004F67 88DC                   <1>         MOV     AH,BL               ; OUTPUT DRIVE NUMBER
00004F69 E85FFFFFFF             <1>         CALL    NEC_OUTPUT
00004F6E 8AA7[41810100]         <1>         MOV     AH, [DSK_TRK+eDI]   ; GET CYLINDER NUMBER
00004F74 E854FFFFFF             <1>         CALL    NEC_OUTPUT
00004F79 E827000000             <1>         CALL    CHK_STAT_2          ; ENDING INTERRUPT AND SENSE STATUS
                                <1>
                                <1> ;----- WAIT FOR HEAD SETTLE
                                <1>
                                <1> DO_WAIT:
00004F7E 9C                     <1>         PUSHF                       ; SAVE STATUS
00004F7F E816FFFFFF             <1>         CALL    HD_WAIT                 ; WAIT FOR HEAD SETTLE TIME
00004F84 9D                     <1>         POPF                        ; RESTORE STATUS
                                <1> RB:
                                <1> NEC_ERR:
                                <1>         ; 08/02/2015 (code trick here from original IBM PC/AT DISKETTE.ASM)
                                <1>         ; (*) nec_err -> retn (push edx -> pop edx) -> nec_err -> retn
00004F85 C3                     <1>         RETn                        ; RETURN TO CALLER
                                <1>
                                <1> ;--------------------------------------------------------------------------
                                <1> ; RECAL
                                <1> ;     RECALIBRATE DRIVE
                                <1> ;
                                <1> ; ON ENTRY:  DI = DRIVE #
                                <1> ;
                                <1> ; ON EXIT:   CY REFLECTS STATUS OF OPERATION.
                                <1> ;--------------------------------------------------------------------------
                                <1> RECAL:
                                <1>         ;PUSH  CX
                                <1>         ; 11/04/2021
00004F86 51                     <1>         push   ecx
00004F87 B8[A34F0000]           <1>         MOV    eAX, RC_BACK         ; LOAD NEC_OUTPUT ERROR
00004F8C 50                     <1>         PUSH   eAX
00004F8D B407                   <1>         MOV    AH,07H               ; RECALIBRATE COMMAND
00004F8F E839FFFFFF             <1>         CALL   NEC_OUTPUT
00004F94 89FB                   <1>         MOV    eBX,eDI              ; BX = DRIVE #
00004F96 88DC                   <1>         MOV    AH,BL
00004F98 E830FFFFFF             <1>         CALL   NEC_OUTPUT           ; OUTPUT THE DRIVE NUMBER
00004F9D E803000000             <1>         CALL   CHK_STAT_2           ; GET THE INTERRUPT AND SENSE INT STATUS
00004FA2 58                     <1>         POP    eAX                  ; THROW AWAY ERROR
                                <1> RC_BACK:
                                <1>         ;POP   CX
                                <1>         ; 11/04/2021
00004FA3 59                     <1>         pop    ecx
00004FA4 C3                     <1>         RETn
                                <1>
                                <1> ;--------------------------------------------------------------------------
                                <1> ; CHK_STAT_2
                                <1> ;     THIS ROUTINE HANDLES THE INTERRUPT RECEIVED AFTER RECALIBRATE,
                                <1> ;     OR SEEK TO THE ADAPTER. THE INTERRUPT IS WAITED FOR, THE
                                <1> ;     INTERRUPT STATUS SENSED, AND THE RESULT RETURNED TO THE CALLER.
                                <1> ;
                                <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
                                <1> ;--------------------------------------------------------------------------
                                <1> CHK_STAT_2:
00004FA5 B8[CD4F0000]           <1>         MOV    eAX, CS_BACK         ; LOAD NEC_OUTPUT ERROR ADDRESS
00004FAA 50                     <1>         PUSH   eAX
00004FAB E828000000             <1>         CALL   WAIT_INT             ; WAIT FOR THE INTERRUPT
00004FB0 721A                   <1>         JC     short J34            ; IF ERROR, RETURN IT
00004FB2 B408                   <1>         MOV    AH,08H               ; SENSE INTERRUPT STATUS COMMAND
00004FB4 E814FFFFFF             <1>         CALL   NEC_OUTPUT
00004FB9 E84A000000             <1>         CALL   RESULTS              ; READ IN THE RESULTS
00004FBE 720C                   <1>         JC     short J34
00004FC0 A0[31810100]           <1>         MOV    AL,[NEC_STATUS]      ; GET THE FIRST STATUS BYTE
00004FC5 2460                   <1>         AND    AL,01100000B         ; ISOLATE THE BITS
00004FC7 3C60                   <1>         CMP    AL,01100000B         ; TEST FOR CORRECT VALUE
00004FC9 7403                   <1>         JZ     short J35            ; IF ERROR, GO MARK IT
00004FCB F8                     <1>         CLC                         ; GOOD RETURN
                                <1> J34:
00004FCC 58                     <1>         POP    eAX                  ; THROW AWAY ERROR RETURN
                                <1> CS_BACK:
00004FCD C3                     <1>         RETn
                                <1> J35:
00004FCE 800D[30810100]40       <1>         OR     byte [DSKETTE_STATUS], BAD_SEEK
00004FD5 F9                     <1>         STC                         ; ERROR RETURN CODE
```

```
5777 00004FD6 EBF4                <1>       JMP    SHORT J34
5778                              <1>
5779                              <1> ;--------------------------------------------------------------------------------
5780                              <1> ; WAIT_INT
5781                              <1> ;     THIS ROUTINE WAITS FOR AN INTERRUPT TO OCCUR A TIME OUT ROUTINE
5782                              <1> ;     TAKES PLACE DURING THE WAIT, SO THAT AN ERROR MAY BE RETURNED
5783                              <1> ;     IF THE DRIVE IS NOT READY.
5784                              <1> ;
5785                              <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
5786                              <1> ;--------------------------------------------------------------------------------
5787                              <1>
5788                              <1> ; 17/12/2014
5789                              <1> ; 2.5 seconds waiting !
5790                              <1> ;(AWARD BIOS - 1999, WAIT_FDU_INT_LOW, WAIT_FDU_INT_HI)
5791                              <1> ; amount of time to wait for completion interrupt from NEC.
5792                              <1>
5793                              <1>
5794                              <1> WAIT_INT:
5795 00004FD8 FB                 <1>       STI                         ; TURN ON INTERRUPTS, JUST IN CASE
5796 00004FD9 F8                 <1>       CLC                         ; CLEAR TIMEOUT INDICATOR
5797                              <1>      ;MOV    BL,10               ; CLEAR THE COUNTERS
5798                              <1>      ;XOR    CX,CX               ; FOR 2 SECOND WAIT
5799                              <1>
5800                              <1>       ; Modification from AWARD BIOS - 1999 (ATORGS.ASM, WAIT
5801                              <1>       ;
5802                              <1>       ;WAIT_FOR_MEM:
5803                              <1>       ;     Waits for a bit at a specified memory location pointed
5804                              <1>       ;     to by ES:[DI] to become set.
5805                              <1>       ;INPUT:
5806                              <1>       ;     AH=Mask to test with.
5807                              <1>       ;     ES:[DI] = memory location to watch.
5808                              <1>       ;     BH:CX=Number of memory refresh periods to delay.
5809                              <1>       ;          (normally 30 microseconds per period.)
5810                              <1>
5811                              <1>       ; waiting for (max.) 2.5 secs in 30 micro units.
5812                              <1> ;     mov    cx, WAIT_FDU_INT_LO       ; 017798
5813                              <1> ;;    mov    bl, WAIT_FDU_INT_HI
5814                              <1> ;     mov    bl, WAIT_FDU_INT_HI + 1
5815                              <1>       ; 27/02/2015
5816 00004FDA B986450100         <1>       mov    ecx, WAIT_FDU_INT_LH      ; 83334 (2.5 seconds)
5817                              <1> WFMS_CHECK_MEM:
5818 00004FDF F605[2D810100]80    <1>       test   byte [SEEK_STATUS],INT_FLAG ; TEST FOR INTERRUPT OCCURRING
5819 00004FE6 7516                <1>       jnz    short J37
5820                              <1> WFMS_HI:
5821 00004FE8 E461                <1>       IN     AL,PORT_B  ; 061h   ; SYS1, wait for lo to hi
5822 00004FEA A810                <1>       TEST   AL,010H                   ; transition on memory
5823 00004FEC 75FA                <1>       JNZ    SHORT WFMS_HI       ; refresh.
5824                              <1> WFMS_LO:
5825 00004FEE E461                <1>       IN     AL,PORT_B           ;SYS1
5826 00004FF0 A810                <1>       TEST   AL,010H
5827 00004FF2 74FA                <1>       JZ     SHORT WFMS_LO
5828 00004FF4 E2E9                <1>       LOOP   WFMS_CHECK_MEM
5829                              <1> ;WFMS_OUTER_LP:
5830                              <1> ;;    or     bl, bl              ; check outer counter
5831                              <1> ;;    jz     short J36A          ; WFMS_TIMEOUT
5832                              <1> ;     dec    bl
5833                              <1> ;     jz     short J36A
5834                              <1> ;     jmp    short WFMS_CHECK_MEM
5835                              <1>
5836                              <1>       ;17/12/2014
5837                              <1>       ;16/12/2014
5838                              <1> ;     mov   byte [wait_count], 0 ; Reset (INT 08H) counter
5839                              <1> ;J36:
5840                              <1> ;     TEST   byte [SEEK_STATUS],INT_FLAG ; TEST FOR INTERRUPT OCCURRING
5841                              <1> ;     JNZ    short J37
5842                              <1>       ;16/12/2014
5843                              <1>       ;LOOP  J36                 ; COUNT DOWN WHILE WAITING
5844                              <1>       ;DEC   BL                  ; SECOND LEVEL COUNTER
5845                              <1>       ;JNZ   short J36
5846                              <1> ;     cmp   byte [wait_count], 46 ; (46/18.2 seconds)
5847                              <1> ;     jb     short J36
5848                              <1>
5849                              <1> ;WFMS_TIMEOUT:
5850                              <1> ;J36A:
5851 00004FF6 800D[30810100]80    <1>       OR     byte [DSKETTE_STATUS], TIME_OUT ; NOTHING HAPPENED
5852 00004FFD F9                  <1>       STC                         ; ERROR RETURN
5853                              <1> J37:
5854 00004FFE 9C                  <1>       PUSHF                       ; SAVE CURRENT CARRY
5855 00004FFF 8025[2D810100]7F    <1>       AND    byte [SEEK_STATUS], ~INT_FLAG ; TURN OFF INTERRUPT FLAG
5856 00005006 9D                  <1>       POPF                        ; RECOVER CARRY
5857 00005007 C3                  <1>       RETn                        ; GOOD RETURN CODE
5858                              <1>
5859                              <1> ;--------------------------------------------------------------------------------
5860                              <1> ; RESULTS
5861                              <1> ;     THIS ROUTINE WILL READ ANYTHING THAT THE NEC CONTROLLER RETURNS
5862                              <1> ;     FOLLOWING AN INTERRUPT.
5863                              <1> ;
5864                              <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
5865                              <1> ;           AX,BX,CX,DX DESTROYED
5866                              <1> ;--------------------------------------------------------------------------------
5867                              <1> RESULTS:
5868 00005008 57                  <1>       PUSH   eDI
5869 00005009 BF[31810100]        <1>       MOV    eDI, NEC_STATUS     ; POINTER TO DATA AREA
5870 0000500E B307                <1>       MOV    BL,7                ; MAX STATUS BYTES
5871 00005010 66BAF403            <1>       MOV    DX,03F4H            ; STATUS PORT
5872                              <1>
5873                              <1> ;-----      WAIT FOR REQUEST FOR MASTER
5874                              <1>
5875                              <1> _R10:
5876                              <1>       ; 16/12/2014
5877                              <1>       ; wait for (max) 0.5 seconds
5878                              <1>       ;MOV   BH,2                ; HIGH ORDER COUNTER
5879                              <1>       ;XOR   CX,CX               ; COUNTER
5880                              <1>
5881                              <1>       ;Time to wait while waiting for each byte of NEC results = .5
```

```
                                <1>       ;seconds.  .5 seconds = 500,000 micros.  500,000/30 = 16,667.
                                <1>       ; 27/02/2015
00005014 B91B410000             <1>       mov    ecx, WAIT_FDU_RESULTS_LH ; 16667
                                <1>       ;mov   cx, WAIT_FDU_RESULTS_LO  ; 16667
                                <1>       ;mov   bh, WAIT_FDU_RESULTS_HI+1 ; 0+1
                                <1>
                                <1> WFPSR_OUTER_LP:
                                <1>       ;
                                <1> WFPSR_CHECK_PORT:
                                <1> J39:                            ; WAIT FOR MASTER
00005019 EC                     <1>       IN     AL,DX              ; GET STATUS
0000501A 24C0                   <1>       AND    AL,11000000B       ; KEEP ONLY STATUS AND DIRECTION
0000501C 3CC0                   <1>       CMP    AL,11000000B       ; STATUS 1 AND DIRECTION 1 ?
0000501E 7418                   <1>       JZ     short J42          ; STATUS AND DIRECTION OK
                                <1> WFPSR_HI:
00005020 E461                   <1>       IN     AL, PORT_B   ;061h  ; SYS1 ; wait for hi to lo
00005022 A810                   <1>       TEST   AL,010H                   ; transition on memory
00005024 75FA                   <1>       JNZ    SHORT WFPSR_HI            ; refresh.
                                <1> WFPSR_LO:
00005026 E461                   <1>       IN     AL, PORT_B          ; SYS1
00005028 A810                   <1>       TEST   AL,010H
0000502A 74FA                   <1>       JZ     SHORT WFPSR_LO
0000502C E2EB                   <1>         LOOP    WFPSR_CHECK_PORT
                                <1>       ;; 27/02/2015
                                <1>       ;;dec  bh
                                <1>       ;;jnz  short WFPSR_OUTER_LP
                                <1>       ;jmp   short WFPSR_TIMEOUT ; fail
                                <1>
                                <1>       ;;mov  byte [wait_count], 0
                                <1> ;J39:                           ; WAIT FOR MASTER
                                <1> ;     IN     AL,DX              ; GET STATUS
                                <1> ;     AND    AL,11000000B       ; KEEP ONLY STATUS AND DIRECTION
                                <1> ;     CMP    AL,11000000B       ; STATUS 1 AND DIRECTION 1 ?
                                <1> ;     JZ     short J42          ; STATUS AND DIRECTION OK
                                <1>       ;LOOP  J39                ; LOOP TILL TIMEOUT
                                <1>       ;DEC   BH                 ; DECREMENT HIGH ORDER COUNTER
                                <1>       ;JNZ   short J39          ; REPEAT TILL DELAY DONE
                                <1>       ;
                                <1>       ;;cmp  byte [wait_count], 10  ; (10/18.2 seconds)
                                <1>       ;;jb   short J39
                                <1>
                                <1> ;WFPSR_TIMEOUT:
0000502E 800D[30810100]80       <1>       OR     byte [DSKETTE_STATUS],TIME_OUT
00005035 F9                     <1>       STC                        ; SET ERROR RETURN
00005036 EB29                   <1>       JMP    SHORT POPRES        ; POP REGISTERS AND RETURN
                                <1>
                                <1> ;-----      READ IN THE STATUS
                                <1>
                                <1> J42:
00005038 EB00                   <1>       JMP    $+2                 ; I/O DELAY
0000503A 6642                   <1>       INC    DX                  ; POINT AT DATA PORT
0000503C EC                     <1>       IN     AL,DX               ; GET THE DATA
                                <1>       ; 16/12/2014
                                <1>       NEWIODELAY
2995 0000503D E6EB              <2>  out 0ebh,al
0000503F 8807                   <1>       MOV    [eDI],AL            ; STORE THE BYTE
00005041 47                     <1>       INC    eDI                 ; INCREMENT THE POINTER
                                <1>       ; 16/12/2014
                                <1> ;     push   cx
                                <1> ;     mov    cx, 30
                                <1> ;wdw2:
                                <1> ;     NEWIODELAY
                                <1> ;     loop   wdw2
                                <1> ;     pop    cx
                                <1>
                                <1>       ;MOV   eCX,3               ; MINIMUM 24 MICROSECONDS FOR NEC
                                <1>       ; 11/04/2021
00005042 B902000000             <1>       mov    ecx, 2
00005047 E85CD3FFFF             <1>       CALL   WAITF               ; WAIT 30 TO 45 MICROSECONDS
0000504C 664A                   <1>       DEC    DX                  ; POINT AT STATUS PORT
0000504E EC                     <1>       IN     AL,DX               ; GET STATUS
                                <1>       ; 16/12/2014
                                <1>       NEWIODELAY
2995 0000504F E6EB              <2>  out 0ebh,al
                                <1>       ;
00005051 A810                   <1>       TEST   AL,00010000B        ; TEST FOR NEC STILL BUSY
00005053 740C                   <1>       JZ     short POPRES        ; RESULTS DONE ?
                                <1>
00005055 FECB                   <1>       DEC    BL                  ; DECREMENT THE STATUS COUNTER
00005057 75BB                   <1>       JNZ   short _R10           ; GO BACK FOR MORE
00005059 800D[30810100]20       <1>       OR     byte [DSKETTE_STATUS],BAD_NEC ; TOO MANY STATUS BYTES
00005060 F9                     <1>       STC                        ; SET ERROR FLAG
                                <1>
                                <1> ;-----      RESULT OPERATION IS DONE
                                <1> POPRES:
00005061 5F                     <1>       POP    eDI
00005062 C3                     <1>       RETn                       ; RETURN WITH CARRY SET
                                <1>
                                <1> ;--------------------------------------------------------------------------------
                                <1> ; READ_DSKCHNG
                                <1> ;     READS THE STATE OF THE DISK CHANGE LINE.
                                <1> ;
                                <1> ; ON ENTRY:  DI = DRIVE #
                                <1> ;
                                <1> ; ON EXIT:   DI = DRIVE #
                                <1> ;            ZF = 0 : DISK CHANGE LINE INACTIVE
                                <1> ;            ZF = 1 : DISK CHANGE LINE ACTIVE
                                <1> ;            AX,CX,DX DESTROYED
                                <1> ;--------------------------------------------------------------------------------
                                <1> READ_DSKCHNG:
00005063 E8A4FDFFFF             <1>       CALL   MOTOR_ON            ; TURN ON THE MOTOR IF OFF
00005068 66BAF703               <1>       MOV    DX,03F7H            ; ADDRESS DIGITAL INPUT REGISTER
0000506C EC                     <1>       IN     AL,DX               ; INPUT DIGITAL INPUT REGISTER
0000506D A880                   <1>       TEST   AL,DSK_CHG          ; CHECK FOR DISK CHANGE LINE ACTIVE
0000506F C3                     <1>       RETn                       ; RETURN TO CALLER WITH ZERO FLAG SET
```

```
                                <1>
                                <1> ;--------------------------------------------------------------------------------
                                <1> ; DRIVE_DET
                                <1> ;     DETERMINES WHETHER DRIVE IS 80 OR 40 TRACKS AND
                                <1> ;     UPDATES STATE INFORMATION ACCORDINGLY.
                                <1> ; ON ENTRY:  DI = DRIVE #
                                <1> ;--------------------------------------------------------------------------------
                                <1> DRIVE_DET:
00005070 E897FDFFFF             <1>       CALL   MOTOR_ON           ; TURN ON MOTOR IF NOT ALREADY ON
00005075 E80CFFFFFF             <1>       CALL   RECAL              ; RECALIBRATE DRIVE
0000507A 724F                   <1>       JC     short DD_BAC       ; ASSUME NO DRIVE PRESENT
0000507C B530                   <1>       MOV    CH,TRK_SLAP        ; SEEK TO TRACK 48
0000507E E884FEFFFF             <1>       CALL   SEEK
00005083 7246                   <1>       JC     short DD_BAC       ; ERROR NO DRIVE
00005085 B50B                   <1>       MOV    CH,QUIET_SEEK+1          ; SEEK TO TRACK 10
                                <1> SK_GIN:
00005087 FECD                   <1>       DEC    CH                 ; DECREMENT TO NEXT TRACK
                                <1>       ;PUSH  CX                 ; SAVE TRACK
                                <1>       ; 11/04/2021
00005089 51                     <1>       push   ecx
0000508A E878FEFFFF             <1>       CALL   SEEK
0000508F 723B                   <1>       JC     short POP_BAC      ; POP AND RETURN
00005091 B8[CC500000]           <1>       MOV    eAX, POP_BAC       ; LOAD NEC OUTPUT ERROR ADDRESS
00005096 50                     <1>       PUSH   eAX
00005097 B404                   <1>       MOV    AH,SENSE_DRV_ST          ; SENSE DRIVE STATUS COMMAND BYTE
00005099 E82FFEFFFF             <1>       CALL   NEC_OUTPUT         ; OUTPUT TO NEC
0000509E 6689F8                 <1>       MOV    AX,DI              ; AL = DRIVE
000050A1 88C4                   <1>       MOV    AH,AL              ; AH = DRIVE
000050A3 E825FEFFFF             <1>       CALL   NEC_OUTPUT         ; OUTPUT TO NEC
000050A8 E85BFFFFFF             <1>       CALL   RESULTS                 ; GO GET STATUS
000050AD 58                     <1>       POP    eAX                ; THROW AWAY ERROR ADDRESS
                                <1>       ;POP   CX                 ; RESTORE TRACK
                                <1>       ; 11/04/2021
000050AE 59                     <1>       pop    ecx
000050AF F605[31810100]10       <1>       TEST   byte [NEC_STATUS], HOME   ; TRACK 0 ?
000050B6 74CF                   <1>       JZ     short SK_GIN       ; GO TILL TRACK 0
000050B8 08ED                   <1>       OR     CH,CH              ; IS HOME AT TRACK 0
000050BA 7408                   <1>       JZ     short IS_80        ; MUST BE 80 TRACK DRIVE
                                <1>
                                <1> ;     DRIVE IS A 360; SET DRIVE TO DETERMINED;
                                <1> ;     SET MEDIA TO DETERMINED AT RATE 250.
                                <1>
000050BC 808F[3D810100]94       <1>       OR     byte [DSK_STATE+eDI], DRV_DET+MED_DET+RATE_250
000050C3 C3                     <1>       RETn                      ; ALL INFORMATION SET
                                <1> IS_80:
000050C4 808F[3D810100]01       <1>       OR     byte [DSK_STATE+eDI], TRK_CAPA ; SETUP 80 TRACK CAPABILITY
                                <1> DD_BAC:
000050CB C3                     <1>       RETn
                                <1> POP_BAC:
                                <1>       ;POP   CX                 ; THROW AWAY
                                <1>       ; 11/04/2021
000050CC 59                     <1>       pop    ecx
000050CD C3                     <1>       RETn
                                <1>
                                <1> fdc_int:
                                <1>         ; 30/07/2015
                                <1>         ; 16/02/2015
                                <1> ;int_0Eh: ; 11/12/2014
                                <1>
                                <1> ;--- HARDWARE INT 0EH -- ( IRQ LEVEL  6 ) -------------------------------------
                                <1> ; DISK_INT
                                <1> ;     THIS ROUTINE HANDLES THE DISKETTE INTERRUPT.
                                <1> ;
                                <1> ; ON EXIT:  THE INTERRUPT FLAG IS SET IN @SEEK_STATUS.
                                <1> ;--------------------------------------------------------------------------------
                                <1> DISK_INT_1:
                                <1>       ;PUSH  AX                 ; SAVE WORK REGISTER
                                <1>       ; 11/04/2021
000050CE 50                     <1>       push   eax
000050CF 1E                     <1>       push   ds
000050D0 66B81000               <1>       mov    ax, KDATA
000050D4 8ED8                   <1>       mov    ds, ax
000050D6 800D[2D810100]80       <1>       OR    byte [SEEK_STATUS], INT_FLAG ; TURN ON INTERRUPT OCCURRED
000050DD B020                   <1>       MOV    AL,EOI             ; END OF INTERRUPT MARKER
000050DF E620                   <1>       OUT    INTA00,AL          ; INTERRUPT CONTROL PORT
000050E1 1F                     <1>       pop    ds
                                <1>       ;POP   AX                 ; RECOVER REGISTER
                                <1>       ; 11/04/2021
000050E2 58                     <1>       pop    eax
000050E3 CF                     <1>       IRETd                     ; RETURN FROM INTERRUPT
                                <1>
                                <1> ;--------------------------------------------------------------------------------
                                <1> ; DSKETTE_SETUP
                                <1> ;     THIS ROUTINE DOES A PRELIMINARY CHECK TO SEE WHAT TYPE OF
                                <1> ;     DISKETTE DRIVES ARE ATTACH TO THE SYSTEM.
                                <1> ;--------------------------------------------------------------------------------
                                <1>
                                <1> ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>
                                <1> DSKETTE_SETUP:
                                <1>       ;PUSH  AX                 ; SAVE REGISTERS
                                <1>       ;PUSH  BX
                                <1>       ;PUSH  CX
000050E4 52                     <1>       PUSH   eDX
                                <1>       ;PUSH  DI
                                <1>       ;;PUSH DS
                                <1>       ; 14/12/2014
                                <1>       ;mov   word [DISK_POINTER], MD_TBL6
                                <1>       ;mov   [DISK_POINTER+2], cs
                                <1>       ;
                                <1>       ;OR    byte [RTC_WAIT_FLAG], 1    ; NO RTC WAIT, FORCE USE OF LOOP
000050E5 31FF                   <1>       XOR    eDI,eDI                   ; INITIALIZE DRIVE POINTER
000050E7 66C705[3D810100]00-    <1>       MOV    WORD [DSK_STATE],0  ; INITIALIZE STATES
000050EF 00                     <1>
000050F0 8025[38810100]33       <1>       AND    byte [LASTRATE],~(STRT_MSK+SEND_MSK) ; CLEAR START & SEND
```

```
6089 000050F7 800D[38810100]C0    <1>       OR     byte [LASTRATE],SEND_MSK ; INITIALIZE SENT TO IMPOSSIBLE
6090 000050FE C605[2D810100]00    <1>       MOV    byte [SEEK_STATUS],0      ; INDICATE RECALIBRATE NEEDED
6091 00005105 C605[2F810100]00    <1>       MOV    byte [MOTOR_COUNT],0      ; INITIALIZE MOTOR COUNT
6092 0000510C C605[2E810100]00    <1>       MOV    byte [MOTOR_STATUS],0     ; INITIALIZE DRIVES TO OFF STATE
6093 00005113 C605[30810100]00    <1>       MOV    byte [DSKETTE_STATUS],0   ; NO ERRORS
6094                              <1>       ;
6095                              <1>       ; 28/02/2015
6096                              <1>       ;mov   word [cfd], 100h
6097 0000511A E868F2FFFF          <1>       call   DSK_RESET
6098 0000511F 5A                  <1>       pop    edx
6099 00005120 F8                  <1>       clc    ; 29/05/2016
6100 00005121 C3                  <1>       retn
6101                              <1>
6102                              <1> ;SUP0:
6103                              <1> ;     CALL   DRIVE_DET           ; DETERMINE DRIVE
6104                              <1> ;     CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
6105                              <1> ;     ; 02/01/2015
6106                              <1> ;     ;INC   DI                  ; POINT TO NEXT DRIVE
6107                              <1> ;     ;CMP   DI,MAX_DRV          ; SEE IF DONE
6108                              <1> ;     ;JNZ   short SUP0          ; REPEAT FOR EACH ORIVE
6109                              <1> ;       cmp     byte [fd1_type], 0
6110                              <1> ;     jna    short sup1
6111                              <1> ;     or     di, di
6112                              <1> ;     jnz    short sup1
6113                              <1> ;     inc    di
6114                              <1> ;       jmp     short SUP0
6115                              <1> ;sup1:
6116                              <1> ;     MOV    byte [SEEK_STATUS],0 ; FORCE RECALIBRATE
6117                              <1> ;     ;AND   byte [RTC_WAIT_FLAG],0FEH ; ALLOW FOR RTC WAIT
6118                              <1> ;     CALL   SETUP_END           ; VARIOUS CLEANUPS
6119                              <1> ;     ;;POP  DS                  ; RESTORE CALLERS REGISTERS
6120                              <1> ;     ;POP   DI
6121                              <1> ;     POP    eDX
6122                              <1> ;     ;POP   CX
6123                              <1> ;     ;POP   BX
6124                              <1> ;     ;POP   AX
6125                              <1> ;     RETn
6126                              <1>
6127                              <1> ;////////////////////////////////////////////////////
6128                              <1> ;; END OF DISKETTE I/O ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
6129                              <1>
6130                              <1> ; 17/04/2021 (TRDOS 386 v2.0.4)
6131                              <1>
6132                              <1> ; 11/04/2021
6133                              <1> ;int13h: ; 21/02/2015
6134                              <1>       ;pushfd
6135                              <1>       ;push  cs
6136                              <1>       ;;call DISK_IO
6137                              <1>       ;;retn
6138                              <1>
6139                              <1> ;;;;;; DISK I/O ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; 21/02/2015 ;;;
6140                              <1> ;////////////////////////////////////////////////////////////////////
6141                              <1>
6142                              <1> ; DISK I/O - Erdogan Tan (Retro UNIX 386 v1 project)
6143                              <1> ; 18/02/2016
6144                              <1> ; 17/02/2016
6145                              <1> ; 23/02/2015
6146                              <1> ; 21/02/2015 (unix386.s)
6147                              <1> ; 22/12/2014 - 14/02/2015 (dsectrm2.s)
6148                              <1> ;
6149                              <1> ; Original Source Code:
6150                              <1> ; DISK ----- 09/25/85 FIXED DISK BIOS
6151                              <1> ; (IBM PC XT Model 286 System BIOS Source Code, 04-21-86)
6152                              <1> ;
6153                              <1> ; Modifications: by reference of AWARD BIOS 1999 (D1A0622)
6154                              <1> ;             Source Code - ATORGS.ASM, AHDSK.ASM
6155                              <1> ;
6156                              <1>
6157                              <1>
6158                              <1> ;The wait for controller to be not busy is 10 seconds.
6159                              <1> ;10,000,000 / 30 = 333,333.  333,333 decimal = 051615h
6160                              <1> ;;WAIT_HDU_CTLR_BUSY_LO   equ    1615h
6161                              <1> ;;WAIT_HDU_CTLR_BUSY_HI   equ      05h
6162                              <1> WAIT_HDU_CTRL_BUSY_LH     equ    51615h ; 21/02/2015
6163                              <1>
6164                              <1> ;The wait for controller to issue completion interrupt is 10 seconds.
6165                              <1> ;10,000,000 / 30 = 333,333.  333,333 decimal = 051615h
6166                              <1> ;;WAIT_HDU_INT_LO  equ    1615h
6167                              <1> ;;WAIT_HDU_INT_HI  equ      05h
6168                              <1> WAIT_HDU_INT_LH    equ    51615h ; 21/02/2015
6169                              <1>
6170                              <1> ;The wait for Data request on read and write longs is
6171                              <1> ;2000 us. (?)
6172                              <1> ;;WAIT_HDU_DRQ_LO  equ    1000   ; 03E8h
6173                              <1> ;;WAIT_HDU_DRQ_HI  equ    0
6174                              <1> WAIT_HDU_DRQ_LH    equ    1000   ; 21/02/2015
6175                              <1>
6176                              <1> ; Port 61h (PORT_B)
6177                              <1> SYS1  equ    61h    ; PORT_B  (diskette.inc)
6178                              <1>
6179                              <1> ; 23/12/2014
6180                              <1> %define CMD_BLOCK  eBP-8  ; 21/02/2015
6181                              <1>
6182                              <1>
6183                              <1> ;--- INT 13H ----------------------------------------------------------------------
6184                              <1> ;                                                                               :
6185                              <1> ; FIXED DISK I/O INTERFACE                                                      :
6186                              <1> ;                                                                               :
6187                              <1> ;     THIS INTERFACE PROVIDES ACCESS TO 5 1/4" FIXED DISKS THROUGH             :
6188                              <1> ;     THE IBM FIXED DISK CONTROLLER.                                           :
6189                              <1> ;                                                                               :
6190                              <1> ;     THE  BIOS  ROUTINES  ARE  MEANT  TO  BE  ACCESSED  THROUGH              :
6191                              <1> ;     SOFTWARE  INTERRUPTS  ONLY.    ANY  ADDRESSES  PRESENT IN               :
6192                              <1> ;     THESE  LISTINGS  ARE  INCLUDED    ONLY  FOR  COMPLETENESS,              :
6193                              <1> ;     NOT  FOR  REFERENCE.  APPLICATIONS   WHICH  REFERENCE  ANY              :
```

```
6194                                <1> ;     ABSOLUTE  ADDRESSES  WITHIN  THE  CODE SEGMENTS  OF  BIOS              :
6195                                <1> ;     VIOLATE  THE  STRUCTURE  AND  DESIGN  OF  BIOS.                        :
6196                                <1> ;                                                                             :
6197                                <1> ;-------------------------------------------------------------------------------
6198                                <1> ;                                                                             :
6199                                <1> ; INPUT  (AH)= HEX COMMAND VALUE                                              :
6200                                <1> ;                                                                             :
6201                                <1> ;     (AH)= 00H  RESET DISK (DL = 80H,81H) / DISKETTE                         :
6202                                <1> ;     (AH)= 01H  READ THE STATUS OF THE LAST DISK OPERATION INTO (AL)         :
6203                                <1> ;                NOTE: DL < 80H - DISKETTE                                    :
6204                                <1> ;                      DL > 80H - DISK                                        :
6205                                <1> ;     (AH)= 02H  READ THE DESIRED SECTORS INTO MEMORY                         :
6206                                <1> ;     (AH)= 03H  WRITE THE DESIRED SECTORS FROM MEMORY                        :
6207                                <1> ;     (AH)= 04H  VERIFY THE DESIRED SECTORS                                   :
6208                                <1> ;     (AH)= 05H  FORMAT THE DESIRED TRACK                                     :
6209                                <1> ;     (AH)= 06H  UNUSED                                                       :
6210                                <1> ;     (AH)= 07H  UNUSED                                                       :
6211                                <1> ;     (AH)= 08H  RETURN THE CURRENT DRIVE PARAMETERS                          :
6212                                <1> ;     (AH)= 09H  INITIALIZE DRIVE PAIR CHARACTERISTICS                        :
6213                                <1> ;                INTERRUPT 41 POINTS TO DATA BLOCK FOR DRIVE 0                :
6214                                <1> ;                INTERRUPT 46 POINTS TO DATA BLOCK FOR DRIVE 1                :
6215                                <1> ;     (AH)= 0AH  READ LONG                                                    :
6216                                <1> ;     (AH)= 0BH  WRITE LONG  (READ & WRITE LONG ENCOMPASS 512 + 4 BYTES ECC)  :
6217                                <1> ;     (AH)= 0CH  SEEK                                                         :
6218                                <1> ;     (AH)= 0DH  ALTERNATE DISK RESET (SEE DL)                                :
6219                                <1> ;     (AH)= 0EH  UNUSED                                                       :
6220                                <1> ;     (AH)= 0FH  UNUSED                                                       :
6221                                <1> ;     (AH)= 10H  TEST DRIVE READY                                             :
6222                                <1> ;     (AH)= 11H  RECALIBRATE                                                  :
6223                                <1> ;     (AH)= 12H  UNUSED                                                       :
6224                                <1> ;     (AH)= 13H  UNUSED                                                       :
6225                                <1> ;     (AH)= 14H  CONTROLLER INTERNAL DIAGNOSTIC                               :
6226                                <1> ;     (AH)= 15H  READ DASD TYPE                                               :
6227                                <1> ;                                                                             :
6228                                <1> ;-------------------------------------------------------------------------------
6229                                <1> ;                                                                             :
6230                                <1> ;     REGISTERS USED FOR FIXED DISK OPERATIONS                                :
6231                                <1> ;                                                                             :
6232                                <1> ;           (DL)   -  DRIVE NUMBER     (80H-81H FOR DISK. VALUE CHECKED)    :
6233                                <1> ;           (DH)   -  HEAD NUMBER          (0-15 ALLOWED, NOT VALUE CHECKED)  :
6234                                <1> ;           (CH)   -  CYLINDER NUMBER  (0-1023, NOT VALUE CHECKED)(SEE CL)    :
6235                                <1> ;           (CL)   -  SECTOR NUMBER    (1-17, NOT VALUE CHECKED)              :
6236                                <1> ;                                                                             :
6237                                <1> ;                     NOTE: HIGH 2 BITS OF CYLINDER NUMBER ARE PLACED         :
6238                                <1> ;                          IN THE HIGH 2 BITS OF THE CL REGISTER              :
6239                                <1> ;                          (10 BITS TOTAL)                                    :
6240                                <1> ;                                                                             :
6241                                <1> ;           (AL)   -  NUMBER OF SECTORS (MAXIMUM POSSIBLE RANGE 1-80H,        :
6242                                <1> ;                                       FOR READ/WRITE LONG 1-79H)            :
6243                                <1> ;                                                                             :
6244                                <1> ;           (ES:BX) -  ADDRESS OF BUFFER FOR READS AND WRITES,                :
6245                                <1> ;                      (NOT REQUIRED FOR VERIFY)                              :
6246                                <1> ;                                                                             :
6247                                <1> ;           FORMAT (AH=5) ES:BX POINTS TO A 512 BYTE BUFFER. THE FIRST        :
6248                                <1> ;                    2*(SECTORS/TRACK) BYTES CONTAIN F,N FOR EACH SECTOR.     :
6249                                <1> ;                    F = 00H FOR A GOOD SECTOR                                :
6250                                <1> ;                        80H FOR A BAD SECTOR                                 :
6251                                <1> ;                    N = SECTOR NUMBER                                        :
6252                                <1> ;                    FOR AN INTERLEAVE OF 2 AND 17 SECTORS/TRACK              :
6253                                <1> ;                    THE TABLE SHOULD BE:                                     :
6254                                <1> ;                                                                             :
6255                                <1> ;              DB    00H,01H,00H,0AH,00H,02H,00H,0BH,00H,03H,00H,0CH          :
6256                                <1> ;              DB    00H,04H,00H,0DH,00H,05H,00H,0EH,00H,06H,00H,0FH          :
6257                                <1> ;              DB    00H,07H,00H,10H,00H,08H,00H,11H,00H,09H                  :
6258                                <1> ;                                                                             :
6259                                <1> ;-------------------------------------------------------------------------------
6260                                <1>
6261                                <1> ;-------------------------------------------------------------------------------
6262                                <1> ; OUTPUT                                                                      :
6263                                <1> ;     AH = STATUS OF CURRENT OPERATION                                        :
6264                                <1> ;          STATUS BITS ARE DEFINED IN THE EQUATES BELOW                       :
6265                                <1> ;     CY = 0 SUCCESSFUL OPERATION (AH=0 ON RETURN)                            :
6266                                <1> ;     CY = 1 FAILED OPERATION (AH HAS ERROR REASON)                           :
6267                                <1> ;                                                                             :
6268                                <1> ;     NOTE:  ERROR 11H  INDICATES THAT THE DATA READ HAD A RECOVERABLE        :
6269                                <1> ;            ERROR WHICH WAS CORRECTED BY THE ECC ALGORITHM.  THE DATA         :
6270                                <1> ;            IS PROBABLY GOOD,   HOWEVER THE BIOS ROUTINE INDICATES AN        :
6271                                <1> ;            ERROR TO ALLOW THE CONTROLLING PROGRAM A CHANCE TO DECIDE         :
6272                                <1> ;            FOR ITSELF.  THE  ERROR  MAY  NOT  RECUR  IF  THE DATA IS        :
6273                                <1> ;            REWRITTEN.                                                       :
6274                                <1> ;                                                                             :
6275                                <1> ;     IF DRIVE PARAMETERS WERE REQUESTED (DL >= 80H),                         :
6276                                <1> ;        INPUT:                                                               :
6277                                <1> ;          (DL) = DRIVE NUMBER                                                :
6278                                <1> ;          ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)                              :
6279                                <1> ;          EBX = Buffer address for fixed disk parameters table (32 bytes)    :
6280                                <1> ;        OUTPUT:                                                              :
6281                                <1> ;          (DL) = NUMBER OF CONSECUTIVE ACKNOWLEDGING DRIVES ATTACHED (1-2)   :
6282                                <1> ;               (CONTROLLER CARD ZERO TALLY ONLY)                           :
6283                                <1> ;          (DH) = MAXIMUM USEABLE VALUE FOR HEAD NUMBER                       :
6284                                <1> ;          (CH) = MAXIMUM USEABLE VALUE FOR CYLINDER NUMBER                   :
6285                                <1> ;          (CL) = MAXIMUM USEABLE VALUE FOR SECTOR NUMBER                     :
6286                                <1> ;               AND CYLINDER NUMBER HIGH BITS                                 :
6287                                <1> ;                                                                             :
6288                                <1> ;     IF READ DASD TYPE WAS REQUESTED,                                        :
6289                                <1> ;                                                                             :
6290                                <1> ;     AH = 0 - NOT PRESENT                                                    :
6291                                <1> ;          1 - DISKETTE - NO CHANGE LINE AVAILABLE                            :
6292                                <1> ;          2 - DISKETTE - CHANGE LINE AVAILABLE                               :
6293                                <1> ;          3 - FIXED DISK                                                     :
6294                                <1> ;                                                                             :
6295                                <1> ;     CX,DX = NUMBER OF 512 BYTE BLOCKS WHEN AH = 3                           :
6296                                <1> ;                                                                             :
6297                                <1> ;     REGISTERS WILL BE PRESERVED EXCEPT WHEN THEY ARE USED TO RETURN         :
6298                                <1> ;     INFORMATION.                                                            :
```

```
6299                            <1> ;                                                                              :
6300                            <1> ;     NOTE: IF AN ERROR IS REPORTED BY THE DISK CODE, THE APPROPRIATE          :
6301                            <1> ;           ACTION IS TO RESET THE DISK, THEN RETRY THE OPERATION.             :
6302                            <1> ;                                                                              :
6303                            <1> ;--------------------------------------------------------------------------------
6304                            <1>
6305                            <1> SENSE_FAIL   EQU    0FFH          ; NOT IMPLEMENTED
6306                            <1> NO_ERR       EQU    0E0H          ; STATUS ERROR/ERROR REGISTER=0
6307                            <1> WRITE_FAULT  EQU    0CCH          ; WRITE FAULT ON SELECTED DRIVE
6308                            <1> UNDEF_ERR    EQU    0BBH          ; UNDEFINED ERROR OCCURRED
6309                            <1> NOT_RDY      EQU    0AAH          ; DRIVE NOT READY
6310                            <1> TIME_OUT     EQU    80H           ; ATTACHMENT FAILED TO RESPOND
6311                            <1> BAD_SEEK     EQU    40H           ; SEEK OPERATION FAILED
6312                            <1> BAD_CNTLR    EQU    20H           ; CONTROLLER HAS FAILED
6313                            <1> DATA_CORRECTED EQU 11H            ; ECC CORRECTED DATA ERROR
6314                            <1> BAD_ECC      EQU    10H           ; BAD ECC ON DISK READ
6315                            <1> BAD_TRACK    EQU    0BH           ; NOT IMPLEMENTED
6316                            <1> BAD_SECTOR   EQU    0AH           ; BAD SECTOR FLAG DETECTED
6317                            <1> ;DMA_BOUNDARY EQU  09H            ; DATA EXTENDS TOO FAR
6318                            <1> INIT_FAIL    EQU    07H           ; DRIVE PARAMETER ACTIVITY FAILED
6319                            <1> BAD_RESET    EQU    05H           ; RESET FAILED
6320                            <1> ;RECORD_NOT_FND EQU 04H           ; REQUESTED SECTOR NOT FOUND
6321                            <1> ;BAD_ADDR_MARK EQU 02H            ; ADDRESS MARK NOT FOUND
6322                            <1> ;BAD_CMD     EQU    01H           ; BAD COMMAND PASSED TO DISK I/O
6323                            <1>
6324                            <1> ;------------------------------------------------------
6325                            <1> ;                                                     :
6326                            <1> ; FIXED DISK PARAMETER TABLE                          :
6327                            <1> ;  -  THE TABLE IS COMPOSED OF A BLOCK DEFINED AS:    :
6328                            <1> ;                                                     :
6329                            <1> ;  +0 (1 WORD) - MAXIMUM NUMBER OF CYLINDERS          :
6330                            <1> ;  +2 (1 BYTE) - MAXIMUM NUMBER OF HEADS              :
6331                            <1> ;  +3 (1 WORD) - NOT USED/SEE PC-XT                   :
6332                            <1> ;  +5 (1 WORD) - STARTING WRITE PRECOMPENSATION CYL   :
6333                            <1> ;  +7 (1 BYTE) - MAXIMUM ECC DATA BURST LENGTH        :
6334                            <1> ;  +8 (1 BYTE) - CONTROL BYTE                         :
6335                            <1> ;                BIT   7 DISABLE RETRIES -OR-         :
6336                            <1> ;                BIT   6 DISABLE RETRIES              :
6337                            <1> ;                BIT   3 MORE THAN 8 HEADS            :
6338                            <1> ;  +9 (3 BYTES)- NOT USED/SEE PC-XT                   :
6339                            <1> ; +12 (1 WORD) - LANDING ZONE                         :
6340                            <1> ; +14 (1 BYTE) - NUMBER OF SECTORS/TRACK              :
6341                            <1> ; +15 (1 BYTE) - RESERVED FOR FUTURE USE              :
6342                            <1> ;                                                     :
6343                            <1> ;      - TO DYNAMICALLY DEFINE A SET OF PARAMETERS    :
6344                            <1> ;        BUILD A TABLE FOR UP TO 15 TYPES AND PLACE   :
6345                            <1> ;        THE CORRESPONDING VECTOR INTO INTERRUPT 41   :
6346                            <1> ;        FOR DRIVE 0 AND INTERRUPT 46 FOR DRIVE 1.    :
6347                            <1> ;                                                     :
6348                            <1> ;-----------------------------------------------------
6349                            <1>
6350                            <1> ;-----------------------------------------------------
6351                            <1> ;                                                     :
6352                            <1> ; HARDWARE SPECIFIC VALUES                            :
6353                            <1> ;                                                     :
6354                            <1> ;  -  CONTROLLER I/O PORT                             :
6355                            <1> ;                                                     :
6356                            <1> ;     > WHEN READ FROM:                               :
6357                            <1> ;     HF_PORT+0 - READ DATA (FROM CONTROLLER TO CPU):
6358                            <1> ;     HF_PORT+1 - GET ERROR REGISTER                  :
6359                            <1> ;     HF_PORT+2 - GET SECTOR COUNT                    :
6360                            <1> ;     HF_PORT+3 - GET SECTOR NUMBER                   :
6361                            <1> ;     HF_PORT+4 - GET CYLINDER LOW                    :
6362                            <1> ;     HF_PORT+5 - GET CYLINDER HIGH (2 BITS)          :
6363                            <1> ;     HF_PORT+6 - GET SIZE/DRIVE/HEAD                 :
6364                            <1> ;     HF_PORT+7 - GET STATUS REGISTER                 :
6365                            <1> ;                                                     :
6366                            <1> ;     > WHEN WRITTEN TO:                              :
6367                            <1> ;     HF_PORT+0 - WRITE DATA (FROM CPU TO CONTROLLER):
6368                            <1> ;     HF_PORT+1 - SET PRECOMPENSATION CYLINDER        :
6369                            <1> ;     HF_PORT+2 - SET SECTOR COUNT                    :
6370                            <1> ;     HF_PORT+3 - SET SECTOR NUMBER                   :
6371                            <1> ;     HF_PORT+4 - SET CYLINDER LOW                    :
6372                            <1> ;     HF_PORT+5 - SET CYLINDER HIGH (2 BITS)          :
6373                            <1> ;     HF_PORT+6 - SET SIZE/DRIVE/HEAD                 :
6374                            <1> ;     HF_PORT+7 - SET COMMAND REGISTER                :
6375                            <1> ;                                                     :
6376                            <1> ;-----------------------------------------------------
6377                            <1>
6378                            <1> ;HF_PORT     EQU    01F0H  ; DISK PORT
6379                            <1> ;HF1_PORT    equ    0170h
6380                            <1> ;HF_REG_PORT EQU    03F6H
6381                            <1> ;HF1_REG_PORT equ   0376h
6382                            <1>
6383                            <1> HDC1_BASEPORT equ   1F0h
6384                            <1> HDC2_BASEPORT equ   170h
6385                            <1>
6386                            <1> align 2
6387                            <1>
6388                            <1> ;----- STATUS REGISTER
6389                            <1>
6390                            <1> ST_ERROR     EQU    00000001B     ;
6391                            <1> ST_INDEX     EQU    00000010B     ;
6392                            <1> ST_CORRCTD   EQU    00000100B     ; ECC CORRECTION SUCCESSFUL
6393                            <1> ST_DRQ       EQU    00001000B     ;
6394                            <1> ST_SEEK_COMPL EQU   00010000B     ; SEEK COMPLETE
6395                            <1> ST_WRT_FLT   EQU    00100000B     ; WRITE FAULT
6396                            <1> ST_READY     EQU    01000000B     ;
6397                            <1> ST_BUSY      EQU    10000000B     ;
6398                            <1>
6399                            <1> ;----- ERROR REGISTER
6400                            <1>
6401                            <1> ERR_DAM      EQU    00000001B     ; DATA ADDRESS MARK NOT FOUND
6402                            <1> ERR_TRK_0    EQU    00000010B     ; TRACK 0 NOT FOUND ON RECAL
6403                            <1> ERR_ABORT    EQU    00000100B     ; ABORTED COMMAND
```

```
                                <1> ;           EQU     00001000B       ; NOT USED
                                <1> ERR_ID      EQU     00010000B       ; ID NOT FOUND
                                <1> ;           EQU     00100000B       ; NOT USED
                                <1> ERR_DATA_ECC EQU    01000000B
                                <1> ERR_BAD_BLOCK EQU   10000000B
                                <1>
                                <1>
                                <1> RECAL_CMD   EQU     00010000B       ; DRIVE RECAL(10H)
                                <1> READ_CMD    EQU     00100000B       ;      READ   (20H)
                                <1> WRITE_CMD   EQU     00110000B       ;      WRITE  (30H)
                                <1> VERIFY_CMD  EQU     01000000B       ;      VERIFY (40H)
                                <1> FMTTRK_CMD  EQU     01010000B       ; FORMAT TRACK      (50H)
                                <1> INIT_CMD    EQU     01100000B       ;   INITIALIZE      (60H)
                                <1> SEEK_CMD    EQU     01110000B       ;      SEEK   (70H)
                                <1> DIAG_CMD    EQU     10010000B       ; DIAGNOSTIC (90H)
                                <1> SET_PARM_CMD EQU    10010001B       ; DRIVE PARMS(91H)
                                <1> NO_RETRIES  EQU     00000001B       ; CHD MODIFIER      (01H)
                                <1> ECC_MODE    EQU     00000010B       ; CMD MODIFIER      (02H)
                                <1> BUFFER_MODE EQU     00001000B       ; CMD MODIFIER      (08H)
                                <1>
                                <1> ;MAX_FILE   EQU     2
                                <1> ;S_MAX_FILE EQU     2
                                <1> MAX_FILE    equ     4               ; 22/12/2014
                                <1> S_MAX_FILE  equ     4               ; 22/12/2014
                                <1>
                                <1> DELAY_1     EQU     25H             ; DELAY FOR OPERATION COMPLETE
                                <1> DELAY_2     EQU     0600H           ; DELAY FOR READY
                                <1> DELAY_3     EQU     0100H           ; DELAY FOR DATA REQUEST
                                <1>
                                <1> HF_FAIL     EQU     08H             ; CMOS FLAG IN BYTE 0EH
                                <1>
                                <1> ;----- COMMAND BLOCK REFERENCE
                                <1>
                                <1> ;CMD_BLOCK  EQU     BP-8            ; @CMD_BLOCK REFERENCES BLOCK HEAD IN SS
                                <1>                                     ;  (BP) POINTS TO COMMAND BLOCK TAIL
                                <1>                                     ;      AS DEFINED BY THE "ENTER" PARMS
                                <1> ; 19/12/2014
                                <1> ORG_VECTOR  equ     4*13h           ; INT 13h vector
                                <1> DISK_VECTOR equ     4*40h           ; INT 40h vector (for floppy disks)
                                <1> ;HDISK_INT  equ     4*76h           ; Primary HDC - Hardware interrupt (IRQ14)
                                <1> ;HDISK_INT1 equ     4*76h           ; Primary HDC - Hardware interrupt (IRQ14)
                                <1> ;HDISK_INT2 equ     4*77h           ; Secondary HDC - Hardware interrupt (IRQ15)
                                <1> ;HF_TBL_VEC equ     4*41h           ; Pointer to 1st fixed disk parameter table
                                <1> ;HF1_TBL_VEC equ    4*46h           ; Pointer to 2nd fixed disk parameter table
                                <1>
                                <1> align 2
                                <1>
                                <1> ;--------------------------------------------------------------
                                <1> ; FIXED DISK I/O SETUP                                          :
                                <1> ;                                                               :
                                <1> ;  -  ESTABLISH TRANSFER VECTORS FOR THE FIXED DISK             :
                                <1> ;  -  PERFORM POWER ON DIAGNOSTICS                              :
                                <1> ;     SHOULD AN ERROR OCCUR A "1701" MESSAGE IS DISPLAYED       :
                                <1> ;                                                               :
                                <1> ;--------------------------------------------------------------
                                <1>
                                <1> ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>
                                <1> DISK_SETUP:
                                <1>       ;CLI
                                <1>       ;;MOV  AX,ABS0                ; GET ABSOLUTE SEGMENT
                                <1>       ;xor   ax,ax
                                <1>       ;MOV   DS,AX                  ; SET SEGMENT REGISTER
                                <1>       ;MOV   AX, [ORG_VECTOR]       ; GET DISKETTE VECTOR
                                <1>       ;MOV   [DISK_VECTOR],AX       ;  INTO INT 40H
                                <1>       ;MOV   AX, [ORG_VECTOR+2]
                                <1>       ;MOV   [DISK_VECTOR+2],AX
                                <1>       ;MOV   word [ORG_VECTOR],DISK_IO ; FIXED DISK HANDLER
                                <1>       ;MOV   [ORG_VECTOR+2],CS
                                <1>       ; 1st controller (primary master, slave)   - IRQ 14
                                <1>       ;;MOV  word [HDISK_INT],HD_INT   ; FIXED DISK INTERRUPT
                                <1>       ;mov   word [HDISK_INT1],HD_INT  ;
                                <1>       ;;MOV  [HDISK_INT+2],CS
                                <1>       ;mov   [HDISK_INT1+2],CS
                                <1>       ; 2nd controller (secondary master, slave) - IRQ 15
                                <1>       ;mov   word [HDISK_INT2],HD1_INT ;
                                <1>       ;mov   [HDISK_INT2+2],CS
                                <1>       ;
                                <1>       ;;MOV  word [HF_TBL_VEC],HD0_DPT ; PARM TABLE DRIVE 80
                                <1>       ;;MOV  word [HF_TBL_VEC+2],DPT_SEGM
                                <1>       ;;MOV  word [HF1_TBL_VEC],HD1_DPT; PARM TABLE DRIVE 81
                                <1>       ;;MOV  word [HF1_TBL_VEC+2],DPT_SEGM
                                <1>       ;push  cs
                                <1>       ;pop   ds
                                <1>       ;mov   word [HDPM_TBL_VEC],HD0_DPT      ; PARM TABLE DRIVE 80h
                                <1>       ;mov   word [HDPM_TBL_VEC+2],DPT_SEGM
00005122 C705[48810100]0000-    <1>       mov    dword [HDPM_TBL_VEC], (DPT_SEGM*16)+HD0_DPT
0000512A 0900                   <1>
                                <1>       ;mov   word [HDPS_TBL_VEC],HD1_DPT      ; PARM TABLE DRIVE 81h
                                <1>       ;mov   word [HDPS_TBL_VEC+2],DPT_SEGM
0000512C C705[4C810100]2000-    <1>       mov    dword [HDPS_TBL_VEC], (DPT_SEGM*16)+HD1_DPT
00005134 0900                   <1>
                                <1>       ;mov   word [HDSM_TBL_VEC],HD2_DPT      ; PARM TABLE DRIVE 82h
                                <1>       ;mov   word [HDSM_TBL_VEC+2],DPT_SEGM
00005136 C705[50810100]4000-    <1>       mov    dword [HDSM_TBL_VEC], (DPT_SEGM*16)+HD2_DPT
0000513E 0900                   <1>
                                <1>       ;mov   word [HDSS_TBL_VEC],HD3_DPT      ; PARM TABLE DRIVE 83h
                                <1>       ;mov   word [HDSS_TBL_VEC+2],DPT_SEGM
00005140 C705[54810100]6000-    <1>       mov    dword [HDSS_TBL_VEC], (DPT_SEGM*16)+HD3_DPT
00005148 0900                   <1>
                                <1>       ;
                                <1>       ;;IN   AL,INTB01              ; TURN ON SECOND INTERRUPT CHIP
                                <1>       ;;;AND AL,0BFH
                                <1>       ;;and  al, 3Fh                ; enable IRQ 14 and IRQ 15
                                <1>       ;;;JMP $+2
```

```
                                <1>       ;;IODELAY
                                <1>       ;;OUT  INTB01,AL
                                <1>       ;;IODELAY
                                <1>       ;;IN   AL,INTA01          ; LET INTERRUPTS PASS THRU TO
                                <1>       ;;AND  AL,0FBH            ;  SECOND CHIP
                                <1>       ;;;JMP $+2
                                <1>       ;;IODELAY
                                <1>       ;;OUT  INTA01,AL
                                <1>       ;
                                <1>       ;STI
                                <1>       ;;PUSH DS                 ; MOVE ABS0 POINTER TO
                                <1>       ;;POP  ES                 ; EXTRA SEGMENT POINTER
                                <1>       ;;;CALL      DDS                ; ESTABLISH DATA SEGMENT
                                <1>       ;;MOV  byte [DISK_STATUS1],0     ; RESET THE STATUS INDICATOR
                                <1>       ;;MOV  byte [HF_NUM],0           ; ZERO NUMBER OF FIXED DISKS
                                <1>       ;;MOV  byte [CONTROL_BYTE],0
                                <1>       ;;MOV  byte [PORT_OFF],0   ; ZERO CARD OFFSET
                                <1>       ; 20/12/2014 - private code by Erdogan Tan
                                <1>                    ; (out of original PC-AT, PC-XT BIOS code)
                                <1>       ;mov   si, hd0_type
0000514A BE[F0680000]           <1>       mov    esi, hd0_type
                                <1>       ;mov   cx, 4
0000514F B904000000             <1>       mov    ecx, 4
                                <1> hde_l:
00005154 AC                     <1>       lodsb
00005155 3C80                   <1>       cmp    al, 80h            ; 8?h = existing
00005157 7206                   <1>       jb     short _L4
00005159 FE05[44810100]         <1>       inc    byte [HF_NUM]      ; + 1 hard (fixed) disk drives
                                <1> _L4: ; 26/02/2015
0000515F E2F3                   <1>       loop   hde_l
                                <1> ;_L4:                           ; 0 <= [HF_NUM] =< 4
                                <1> ;L4:
                                <1>       ;
                                <1>       ;; 31/12/2014 - cancel controller diagnostics here
                                <1>       ;;;mov       cx, 3  ; 26/12/2014 (Award BIOS 1999)
                                <1>       ;;mov  cl, 3
                                <1>       ;;
                                <1>       ;;MOV  DL,80H             ; CHECK THE CONTROLLER
                                <1> ;;hdc_dl:
                                <1>       ;;MOV  AH,14H             ; USE CONTROLLER DIAGNOSTIC COMMAND
                                <1>       ;;INT  13H                ; CALL BIOS WITH DIAGNOSTIC COMMAND
                                <1>       ;;;JC  short CTL_ERRX           ; DISPLAY ERROR MESSAGE IF BAD RETURN
                                <1>       ;;;jc  short POD_DONE ;22/12/2014
                                <1>       ;;jnc  short hdc_reset0
                                <1>       ;;loop hdc_dl
                                <1>       ;;; 27/12/2014
                                <1>       ;;stc
                                <1>       ;;retn
                                <1>       ;
                                <1> ;;hdc_reset0:
                                <1>       ; 18/01/2015
00005161 8A0D[44810100]         <1>       mov    cl, [HF_NUM]
00005167 20C9                   <1>       and    cl, cl
00005169 740D                   <1>       jz     short POD_DONE
                                <1>       ;
0000516B B27F                   <1>       mov    dl, 7Fh
                                <1> hdc_reset1:
0000516D FEC2                   <1>       inc    dl
                                <1>       ;; 31/12/2015
                                <1>       ;;push dx
                                <1>       ;;push cx
                                <1>       ;;push ds
                                <1>       ;;sub  ax, ax
                                <1>       ;;mov  ds, ax
                                <1>       ;;MOV  AX, [TIMER_LOW]     ; GET START TIMER COUNTS
                                <1>       ;;pop  ds
                                <1>       ;;MOV  BX,AX
                                <1>       ;;ADD  AX,6*182           ; 60 SECONDS* 18.2
                                <1>       ;;MOV  CX,AX
                                <1>       ;;mov  word [wait_count], 0 ; 22/12/2014 (reset wait counter)
                                <1>       ;;
                                <1>       ;; 31/12/2014 - cancel HD RESET 1
                                <1>       ;;CALL HD_RESET_1         ; SET UP DRIVE 0, (1,2,3)
                                <1>       ;;pop  cx
                                <1>       ;;pop  dx
                                <1>       ;;
                                <1>       ; 18/01/2015
0000516F B40D                   <1>       mov    ah, 0Dh ; ALTERNATE RESET
                                <1>       ;int   13h
00005171 E8ED000000             <1>       call   int13h
00005176 E2F5                   <1>       loop   hdc_reset1
                                <1>       ;clc   ; 29/05/2016
                                <1> POD_DONE:
00005178 C3                     <1>       RETn
                                <1>
                                <1> ;;-----     POD_ERROR
                                <1>
                                <1> ;;CTL_ERRX:
                                <1> ;     ;MOV   SI,OFFSET F1782     ; CONTROLLER ERROR
                                <1> ;     ;CALL  SET_FAIL            ; DO NOT IPL FROM DISK
                                <1> ;     ;CALL  E_MSG               ; DISPLAY ERROR AND SET (BP) ERROR FLAG
                                <1> ;     ;JMP   short POD_DONE
                                <1>
                                <1> ;;HD_RESET_1:
                                <1> ;;    ;PUSH  BX                  ; SAVE TIMER LIMITS
                                <1> ;;    ;PUSH  CX
                                <1> ;;RES_1: MOV AH,09H              ; SET DRIVE PARAMETERS
                                <1> ;;    INT    13H
                                <1> ;;    JC     short RES_2
                                <1> ;;    MOV    AH,11H              ; RECALIBRATE DRIVE
                                <1> ;;    INT    13H
                                <1> ;;    JNC    short RES_CK        ; DRIVE OK
                                <1> ;;RES_2: ;CALL POD_TCHK          ; CHECK TIME OUT
                                <1> ;;    cmp    word [wait_count], 6*182 ; waiting time (in timer ticks)
                                <1> ;;                              ; (30 seconds)
```

```
6610                    <1> ;;    ;cmc
6611                    <1> ;;    ;JNC   short RES_1
6612                    <1> ;;    jb     short RES_1
6613                    <1> ;;;RES_FL: ;MOV SI,OFFSET F1781 ; INDICATE DISK 1 FAILURE;
6614                    <1> ;;    ;TEST  DL,1
6615                    <1> ;;    ;JNZ   RES_E1
6616                    <1> ;;    ;MOV   SI,OFFSET F1780     ; INDICATE DISK 0 FAILURE
6617                    <1> ;;    ;CALL  SET_FAIL            ; DO NOT TRY TO IPL DISK 0
6618                    <1> ;;    ;JMP   SHORT RES_E1
6619                    <1> ;;RES_ER: ; 22/12/2014
6620                    <1> ;;RES_OK:
6621                    <1> ;;    ;POP   CX                  ; RESTORE TIMER LIMITS
6622                    <1> ;;    ;POP   BX
6623                    <1> ;;    RETn
6624                    <1> ;;
6625                    <1> ;;RES_RS: MOV AH,00H             ; RESET THE DRIVE
6626                    <1> ;;    INT    13H
6627                    <1> ;;RES_CK: MOV AH,08H             ; GET MAX CYLINDER,HEAD,SECTOR
6628                    <1> ;;    MOV    BL,DL               ; SAVE DRIVE CODE
6629                    <1> ;;    INT    13H
6630                    <1> ;;    JC     short RES_ER
6631                    <1> ;;    MOV    [NEC_STATUS],CX     ; SAVE MAX CYLINDER, SECTOR
6632                    <1> ;;    MOV    DL,BL               ; RESTORE DRIVE CODE
6633                    <1> ;;RES_3: MOV AX,0401H            ; VERIFY THE LAST SECTOR
6634                    <1> ;;    INT    13H
6635                    <1> ;;    JNC    short RES_OK        ; VERIFY OK
6636                    <1> ;;    CMP    AH,BAD_SECTOR       ; OK ALSO IF JUST ID READ
6637                    <1> ;;    JE     short RES_OK
6638                    <1> ;;    CMP    AH,DATA_CORRECTED
6639                    <1> ;;    JE     short RES_OK
6640                    <1> ;;    CMP    AH,BAD_ECC
6641                    <1> ;;    JE     short RES_OK
6642                    <1> ;;    ;CALL  POD_TCHK            ; CHECK FOR TIME OUT
6643                    <1> ;;    cmp    word [wait_count], 6*182 ; waiting time (in timer ticks)
6644                    <1> ;;                                ; (60 seconds)
6645                    <1> ;;    cmc
6646                    <1> ;;    JC     short RES_ER        ; FAILED
6647                    <1> ;;    MOV    CX,[NEC_STATUS]     ; GET SECTOR ADDRESS, AND CYLINDER
6648                    <1> ;;    MOV    AL,CL               ; SEPARATE OUT SECTOR NUMBER
6649                    <1> ;;    AND    AL,3FH
6650                    <1> ;;    DEC    AL                  ; TRY PREVIOUS ONE
6651                    <1> ;;    JZ     short RES_RS        ; WE'VE TRIED ALL SECTORS ON TRACK
6652                    <1> ;;    AND    CL,0C0H             ; KEEP CYLINDER BITS
6653                    <1> ;;    OR     CL,AL               ; MERGE SECTOR WITH CYLINDER BITS
6654                    <1> ;;    MOV    [NEC_STATUS],CX     ; SAVE CYLINDER, NEW SECTOR NUMBER
6655                    <1> ;;    JMP    short RES_3         ; TRY AGAIN
6656                    <1> ;;;RES_ER: MOV SI,OFFSET F1791   ; INDICATE DISK 1 ERROR
6657                    <1> ;;    ;TEST  DL,1
6658                    <1> ;;    ;JNZ   short RES_E1
6659                    <1> ;;    ;MOV   SI,OFFSET F1790     ; INDICATE DISK 0 ERROR
6660                    <1> ;;;RES_E1:
6661                    <1> ;;    ;CALL  E_MSG               ; DISPLAY ERROR AND SET (BP) ERROR FLAG
6662                    <1> ;;;RES_OK:
6663                    <1> ;;    ;POP   CX                  ; RESTORE TIMER LIMITS
6664                    <1> ;;    ;POP   BX
6665                    <1> ;;    ;RETn
6666                    <1> ;
6667                    <1> ;;SET_FAIL:
6668                    <1> ;     ;MOV   AX,X*(CMOS_DIAG+NMI) ; GET CMOS ERROR BYTE
6669                    <1> ;     ;CALL  CMOS_READ
6670                    <1> ;     ;OR    AL,HF_FAIL          ; SET DO NOT IPL FROM DISK FLAG
6671                    <1> ;     ;XCHG  AH,AL               ; SAVE IT
6672                    <1> ;     ;CALL  CMOS_WRITE          ; PUT IT OUT
6673                    <1> ;     ;RETn
6674                    <1> ;
6675                    <1> ;;POD_TCHK:                      ; CHECK FOR 30 SECOND TIME OUT
6676                    <1> ;     ;POP   AX                  ; SAVE RETURN
6677                    <1> ;     ;POP   CX                  ; GET TIME OUT LIMITS
6678                    <1> ;     ;POP   BX
6679                    <1> ;     ;PUSH  BX                  ; AND SAVE THEM AGAIN
6680                    <1> ;     ;PUSH  CX
6681                    <1> ;     ;PUSH  AX
6682                    <1> ;     ;push  ds
6683                    <1> ;     ;xor   ax, ax
6684                    <1> ;     ;mov   ds, ax              ; RESTORE RETURN
6685                    <1> ;     ;MOV   AX, [TIMER_LOW]     ; AX = CURRENT TIME
6686                    <1> ;     ;                          ; BX = START TIME
6687                    <1> ;     ;                          ; CX = END TIME
6688                    <1> ;     ;pop   ds
6689                    <1> ;     ;CMP   BX,CX
6690                    <1> ;     ;JB    short TCHK1         ; START < END
6691                    <1> ;     ;CMP   BX,AX
6692                    <1> ;     ;JB    short TCHKG         ; END < START < CURRENT
6693                    <1> ;     ;JMP   SHORT TCHK2         ; END, CURRENT < START
6694                    <1> ;;TCHK1: CMP AX,BX
6695                    <1> ;;    JB     short TCHKNG        ; CURRENT < START < END
6696                    <1> ;;TCHK2: CMP AX,CX
6697                    <1> ;;    JB     short TCHKG         ; START < CURRENT < END
6698                    <1> ;;                               ; OR CURRENT < END < START
6699                    <1> ;;TCHKNG: STC                    ; CARRY SET INDICATES TIME OUT
6700                    <1> ;;    RETn
6701                    <1> ;;TCHKG: CLC                     ; INDICATE STILL TIME
6702                    <1> ;;    RETn
6703                    <1> ;;
6704                    <1> ;;int_13h:
6705                    <1>
6706                    <1> ;-----------------------------------------
6707                    <1> ;     FIXED DISK BIOS ENTRY POINT      :
6708                    <1> ;-----------------------------------------
6709                    <1>
6710                    <1> ; 15/01/2017
6711                    <1> ; 14/01/2017
6712                    <1> ; 07/01/2017
6713                    <1> ; 02/01/2017
6714                    <1> ; 01/06/2016
```

```
                                <1> ; 16/05/2016, 27/05/2016, 28/05/2016, 29/05/2016
                                <1> ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                                <1> int33h:  ; DISK I/O
                                <1>       ; 29/05/2016
00005179 80642408FE             <1>       and   byte [esp+8], 11111110b  ; clear carry bit of eflags register
                                <1>       ; 16/05/2016
0000517E 1E                     <1>       push  ds
0000517F 53                     <1>       push  ebx ; user's buffer address (virtual)
00005180 66BB1000               <1>       mov   bx, KDATA ; System (Kernel's) data segment
00005184 8EDB                   <1>       mov   ds, bx
                                <1>
                                <1>       ;;15/01/2017
                                <1>       ; 14/01/2017
                                <1>       ; 02/01/2017
                                <1>       ;;mov  byte [intflg], 33h  ; disk io interrupt
                                <1>       ;pop  ebx
                                <1>       ;mov  [user_buffer], ebx
                                <1>
00005186 8F05[348D0100]         <1>       pop   dword [user_buffer] ; 01/06/2016
                                <1>
0000518C C605[6E860100]00       <1>       mov   byte [scount], 0 ; sector count for transfer
00005193 80FC03                 <1>       cmp   ah, 03h ; chs write
00005196 7744                   <1>       ja    short int33h_2
00005198 7407                   <1>       je    short int33h_0
0000519A 80FC02                 <1>       cmp   ah, 02h ; chs read
0000519D 726A                   <1>       jb    short int33h_5
0000519F EB63                   <1>       jmp   short int33h_4
                                <1> int33h_0:
                                <1>       ; transfer user's buffer content to sector buffer
000051A1 51                     <1>       push  ecx
000051A2 0FB6C8                 <1>       movzx ecx, al
                                <1> int33h_1:
000051A5 56                     <1>       push  esi
000051A6 8B35[348D0100]         <1>       mov   esi, [user_buffer]
                                <1>       ; esi = user's buffer address (virtual, ebx)
000051AC 57                     <1>       push  edi
000051AD 06                     <1>       push  es
000051AE 50                     <1>       push  eax
000051AF 66B81000               <1>       mov   ax, KDATA
000051B3 8EC0                   <1>       mov   es, ax
000051B5 BF00000700             <1>       mov   edi, Cluster_Buffer
000051BA C1E109                 <1>       shl   ecx, 9 ; * 512
000051BD E8CAC30000             <1>       call  transfer_from_user_buffer
000051C2 58                     <1>       pop   eax
000051C3 07                     <1>       pop   es
000051C4 5F                     <1>       pop   edi
000051C5 5E                     <1>       pop   esi
000051C6 59                     <1>       pop   ecx
000051C7 7340                   <1>       jnc   short int33h_5
000051C9 8B1D[348D0100]         <1>       mov   ebx, [user_buffer] ; 01/06/2016
000051CF 1F                     <1>       pop   ds
                                <1>
                                <1>       ;;15/01/2017
                                <1>       ; 02/01/2017
                                <1>       ;cli
                                <1>       ;;mov  byte [ss:intflg], 0 ; 07/01/2017
                                <1>       ;
                                <1>       ; (*) 29/05/2016
                                <1>       ; (*) retf 4 ; skip eflags on stack
                                <1>
                                <1>       ; 29/05/2016 -set carry flag on stack-
                                <1>       ; [esp] = EIP
                                <1>       ; [esp+4] = CS
                                <1>       ; [esp+8] = E-FLAGS
000051D0 804C240801             <1>       or    byte [esp+8], 1  ; set carry bit of eflags register
                                <1>       ; [esp+12] = ESP (user)
                                <1>       ; [esp+16] = SS (User)
000051D5 B8FF000000             <1>       mov   eax, 0FFh ; Unknown error !?
                                <1>       ;iretd
000051DA EB79                   <1>       jmp   short int33h_7  ; 07/01/2017
                                <1>
                                <1>       ; (*) 29/05/2016 - 'ref 4' intruction causes to stack fault
                                <1>       ; (OUTER-PRIVILEGE-LEVEL)
                                <1>       ; INTEL 80386 PROGRAMMER'S REFERENCE MANUAL 1986
                                <1>       ; // RETF instruction:
                                <1>       ;
                                <1>       ; IF OperandMode=32 THEN
                                <1>       ;    Load CS:EIP from stack;
                                <1>       ;    Set CS RPL to CPL;
                                <1>       ;    Increment eSP by 8 plus the immediate offset if it exists;
                                <1>       ;    Load SS:eSP from stack;
                                <1>       ; ELSE (* OperandMode=16 *)
                                <1>       ;    Load CS:IP from stack;
                                <1>       ;    Set CS RPL to CPL;
                                <1>       ;    Increment eSP by 4 plus the immediate offset if it exists;
                                <1>       ;    Load SS:eSP from stack;
                                <1>       ; FI;
                                <1>       ;
                                <1>       ; //
                                <1>
                                <1> int33h_2:
000051DC 80FC05                 <1>       cmp   ah, 05h ; format track
000051DF 770A                   <1>       ja    short int33h_3
000051E1 7226                   <1>       jb    short int33h_5
000051E3 51                     <1>       push  ecx
000051E4 B901000000             <1>       mov   ecx, 1
000051E9 EBBA                   <1>       jmp   short int33h_1
                                <1> int33h_3:
000051EB 80FC1C                 <1>       cmp   ah, 1Ch ; LBA write
000051EE 7719                   <1>       ja    short int33h_5
000051F0 74AF                   <1>       je    short int33h_0
000051F2 80FC1B                 <1>       cmp   ah, 1Bh ; LBA read
000051F5 740D                   <1>       je    short int33h_4
000051F7 80FC08                 <1>       cmp   ah, 08h ; get disk parameters
000051FA 750D                   <1>       jne   short int33h_5
```

```
                                <1>       ; 01/06/2016
000051FC 8B1D[348D0100]         <1>       mov   ebx, [user_buffer] ; user's buffer address
00005202 EB0A                   <1>       jmp   short int33h_6
                                <1> int33h_4:
00005204 A2[6E860100]           <1>       mov   byte [scount], al ; <= 128 sectors
                                <1> int33h_5:
00005209 BB00000700             <1>       mov   ebx, Cluster_Buffer ; max. 65536 bytes
                                <1>                                 ; buf. addr: 70000h
                                <1>       ;mov  byte [ClusterBuffer_Valid], 0
                                <1> int33h_6:
0000520E 1F                     <1>       pop   ds
0000520F 9C                     <1>       pushfd
00005210 0E                     <1>       push  cs
00005211 E856000000             <1>       call  DISK_IO
00005216 2E8B1D[348D0100]       <1>       mov   ebx, [CS:user_buffer] ; 01/06/2016
0000521D 723D                   <1>       jc    short int33h_9
                                <1>       ;
0000521F 2E803D[6E860100]00     <1>       cmp   byte [CS:scount], 0
00005227 762C                   <1>       jna   short int33h_7
                                <1>       ; transfer sector buffer content to user's buffer
00005229 06                     <1>       push  es
0000522A 1E                     <1>       push  ds
0000522B 50                     <1>       push  eax
0000522C 66B81000               <1>       mov   ax, KDATA
00005230 8ED8                   <1>       mov   ds, ax
00005232 8EC0                   <1>       mov   es, ax
00005234 51                     <1>       push  ecx
00005235 56                     <1>       push  esi
00005236 57                     <1>       push  edi
00005237 0FB60D[6E860100]       <1>       movzx ecx, byte [scount]
0000523E C1E109                 <1>       shl   ecx, 9 ; * 512 bytes
00005241 89DF                   <1>       mov   edi, ebx ; user's buffer address
00005243 BE00000700             <1>       mov   esi, Cluster_Buffer
00005248 E8F5C20000             <1>       call  transfer_to_user_buffer
0000524D 5F                     <1>       pop   edi
0000524E 5E                     <1>       pop   esi
0000524F 59                     <1>       pop   ecx
00005250 58                     <1>       pop   eax
00005251 1F                     <1>       pop   ds
00005252 07                     <1>       pop   es
00005253 7202                   <1>       jc    short int33h_8
                                <1> int33h_7:
00005255 FA                     <1>       cli
                                <1>       ;;15/01/2017
                                <1>       ;;mov  byte [ss:intflg], 0 ; 07/01/2017
                                <1>       ; cf = 0  ; use eflags which is in stack
00005256 CF                     <1>       iretd
                                <1> int33h_8:
00005257 B8FF000000             <1>       mov   eax, 0FFh ; Unknown error !?
                                <1> int33h_9:
                                <1>       ; cf = 1
                                <1>
                                <1>       ; (*) 29/05/2016
                                <1>       ; (*) retf 4 ; skip eflags on stack
                                <1>       ; Note: This 'retf 4' was wrong, -it was causing
                                <1>       ;       to stack errors in ring 3-
                                <1>       ;     POP sequence of 'retf 4' is as
                                <1>       ;       "eip, cs, eflags, esp, ss, +4 bytes"
                                <1>         ;       it is not as "eip, cs, +4 bytes, esp, ss" !
                                <1>
                                <1>       ; 29/05/2016 -set carry flag on stack-
0000525C 804C240801             <1>       or    byte [esp+8], 1  ; set carry bit of eflags register
                                <1>       ;iretd
00005261 EBF2                   <1>       jmp   short int33h_7 ; 07/01/2017
                                <1>
                                <1> ; 11/04/2021
                                <1> int13h: ; 21/02/2015
00005263 F8                     <1>       clc ; 11/04/2021
00005264 9C                     <1>       pushfd
00005265 0E                     <1>       push  cs
00005266 E801000000             <1>       call  DISK_IO
0000526B C3                     <1>       retn
                                <1>
                                <1> ; 18/04/2021 - TRDOS 386 v2.0.4
                                <1> ; 11/04/2021 - TRDOS 386 v2.0.3
                                <1> ; 30/08/2020
                                <1> ; 09/12/2017
                                <1> ; 29/05/2016
                                <1> ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>
                                <1> DISK_IO:
0000526C 80FA80                 <1>       CMP   DL,80H              ; TEST FOR FIXED DISK DRIVE
                                <1>       ;JAE  short A1            ; YES, HANDLE HERE
                                <1>       ;;;INT 40H                ; DISKETTE HANDLER
                                <1>       ;;call int40h
                                <1>       ;jb   DISKETTE_IO_1
                                <1> ;RET_2:
                                <1>       ;RETf 2                   ; BACK TO CALLER
                                <1> ;     retf  4
                                <1>       ; 11/04/2021
0000526F 7305                   <1>       jnb   short A1
00005271 E941F0FFFF             <1>       jmp   DISKETTE_IO_1
                                <1> A1:
00005276 FB                     <1>       STI                       ; ENABLE INTERRUPTS
                                <1>       ;; 04/01/2015
                                <1>       ;;OR  AH,AH
                                <1>       ;;JNZ short A2
                                <1>       ;;INT 40H                 ; RESET NEC WHEN AH=0
                                <1>       ;;SUB AH,AH
00005277 80FA83                 <1>       CMP   DL,(80H + S_MAX_FILE - 1)
                                <1>       ;JA   short RET_2
0000527A 7614                   <1>       jna   short _A0
                                <1>       ; 29/05/2016
0000527C 1E                     <1>       push  ds
                                <1>       ;push ax
```

```
6925                                    <1>       ; 11/04/2021
6926 0000527D 50                        <1>       push   eax
6927 0000527E 66B81000                  <1>       mov    ax, KDATA
6928 00005282 8ED8                      <1>       mov    ds, ax
6929                                    <1>       ;pop   ax
6930                                    <1>       ; 11/04/2021
6931 00005284 58                        <1>       pop    eax
6932 00005285 B4AA                      <1>       mov    ah, 0AAh     ; Hard disk drive not ready !
6933                                    <1>                           ; (Programmer's guide to AMIBIOS, 1992)
6934 00005287 8825[43810100]            <1>       mov    byte [DISK_STATUS1], ah
6935 0000528D 1F                        <1>       pop    ds
6936 0000528E EB38                      <1>       jmp    short RET_2
6937                                    <1> _A0:
6938                                    <1>       ; 18/01/2015
6939 00005290 08E4                      <1>       or     ah,ah
6940 00005292 743A                      <1>       jz     short A4
6941 00005294 80FC0D                    <1>       cmp    ah, 0Dh      ; Alternate reset
6942 00005297 7504                      <1>       jne    short A2
6943 00005299 28E4                      <1>       sub    ah,ah  ; Reset
6944 0000529B EB31                      <1>       jmp    short A4
6945                                    <1> A2:
6946 0000529D 80FC08                    <1>       CMP    AH,08H              ; GET PARAMETERS IS A SPECIAL CASE
6947                                    <1>       ;JNZ   short A3
6948                                    <1>       ;JMP   GET_PARM_N
6949 000052A0 0F841E030000              <1>       je     GET_PARM_N
6950 000052A6 80FC15                    <1> A3:   CMP    AH,15H              ; READ DASD TYPE IS ALSO
6951                                    <1>       ;JNZ   short A4
6952                                    <1>       ;JMP   READ_DASD_TYPE
6953 000052A9 0F84D0020000              <1>       je     READ_DASD_TYPE
6954                                    <1>       ; 02/02/2015
6955 000052AF 80FC1D                    <1>       cmp    ah, 1Dh             ; (Temporary for Retro UNIX 386 v1)
6956                                    <1>       ; 12/01/2015
6957 000052B2 F5                        <1>       cmc
6958 000052B3 7319                      <1>       jnc    short A4
6959                                    <1> int33h_bad_cmd:
6960                                    <1>       ; 16/05/2016
6961                                    <1>       ; 30/01/2015
6962                                    <1>       ; 29/05/2016
6963 000052B5 1E                        <1>       push   ds
6964 000052B6 6650                      <1>       push   ax
6965 000052B8 66B81000                  <1>       mov    ax, KDATA
6966 000052BC 8ED8                      <1>       mov    ds, ax
6967 000052BE 6658                      <1>       pop    ax
6968 000052C0 B401                      <1>       mov    ah, BAD_CMD
6969 000052C2 8825[43810100]            <1>       mov     [DISK_STATUS1], ah ; BAD_CMD  ; COMMAND ERROR
6970                                    <1>         ;jmp short RET_2
6971                                    <1> RET_2:
6972                                    <1>       ; (*) 29/05/2016
6973                                    <1>       ; (*) retf 4
6974 000052C8 804C240801                <1>       or     byte [esp+8], 1 ; set carry bit of eflags register
6975 000052CD CF                        <1>       iretd
6976                                    <1> A4:                           ; SAVE REGISTERS DURING OPERATION
6977 000052CE C8080000                  <1>       ENTER  8,0              ; SAVE (BP) AND MAKE ROOM FOR @CMD_BLOCK
6978 000052D2 53                        <1>       PUSH   eBX              ;  IN THE STACK, THE COMMAND BLOCK IS:
6979 000052D3 51                        <1>       PUSH   eCX              ;   @CMD_BLOCK == BYTE PTR [BP]-8
6980 000052D4 52                        <1>       PUSH   eDX
6981 000052D5 1E                        <1>       PUSH   DS
6982 000052D6 06                        <1>       PUSH   ES
6983 000052D7 56                        <1>       PUSH   eSI
6984 000052D8 57                        <1>       PUSH   eDI
6985                                    <1>       ;;04/01/2015
6986                                    <1>       ;;OR   AH,AH              ; CHECK FOR RESET
6987                                    <1>       ;;JNZ  short A5
6988                                    <1>       ;;MOV  DL,80H             ; FORCE DRIVE 80 FOR RESET
6989                                    <1> ;;A5:
6990                                    <1>       ;push  cs
6991                                    <1>       ;pop   ds
6992                                    <1>       ; 21/02/2015
6993                                    <1>       ;push  ax
6994                                    <1>       ; 11/04/2021
6995 000052D9 50                        <1>       push   eax
6996 000052DA 66B81000                  <1>       mov    ax, KDATA
6997 000052DE 8ED8                      <1>       mov    ds, ax
6998 000052E0 8EC0                      <1>       mov    es, ax
6999                                    <1>       ;pop   ax
7000                                    <1>       ; 11/04/2021
7001 000052E2 58                        <1>       pop    eax
7002 000052E3 E88D000000                <1>       CALL   DISK_IO_CONT       ; PERFORM THE OPERATION
7003                                    <1>       ;;CALL DDS                ; ESTABLISH SEGMENT
7004 000052E8 8A25[43810100]            <1>       MOV    AH,[DISK_STATUS1]  ; GET STATUS FROM OPERATION
7005                                    <1>       ;(*) CMP AH,1             ; SET THE CARRY FLAG TO INDICATE
7006                                    <1>       ;(*) CMC                  ; SUCCESS OR FAILURE
7007 000052EE 5F                        <1>       POP    eDI                ; RESTORE REGISTERS
7008 000052EF 5E                        <1>       POP    eSI
7009 000052F0 07                        <1>       POP    ES
7010 000052F1 1F                        <1>       POP    DS
7011 000052F2 5A                        <1>       POP    eDX
7012 000052F3 59                        <1>       POP    eCX
7013 000052F4 5B                        <1>       POP    eBX
7014 000052F5 C9                        <1>       LEAVE                     ; ADJUST (SP) AND RESTORE (BP)
7015                                    <1>       ;RETf  2                  ; THROW AWAY SAVED FLAGS
7016                                    <1>       ; (*) 29/05/2016
7017                                    <1>       ; (*) retf 4
7018 000052F6 80FC01                    <1>       cmp    ah, 1
7019 000052F9 7205                      <1>       jc     short _A5
7020 000052FB 804C240801                <1>       or     byte [esp+8], 1 ; set carry bit of eflags register
7021                                    <1> _A5:
7022 00005300 CF                        <1>       iretd
7023                                    <1>
7024                                    <1> ; 21/02/2015
7025                                    <1> ;       dw --> dd
7026                                    <1> D1:                           ; FUNCTION TRANSFER TABLE
7027 00005301 [BB540000]                <1>       dd     DISK_RESET         ; 000H
7028 00005305 [32550000]                <1>       dd     RETURN_STATUS      ; 001H
7029 00005309 [3F550000]                <1>       dd     DISK_READ          ; 002H
```

```
7030 0000530D [48550000]          <1>       dd     DISK_WRITE          ; 003H
7031 00005311 [51550000]          <1>       dd     DISK_VERF           ; 004H
7032 00005315 [69550000]          <1>       dd     FMT_TRK             ; 005H
7033 00005319 [B1540000]          <1>       dd     BAD_COMMAND         ; 006H FORMAT BAD SECTORS
7034 0000531D [B1540000]          <1>       dd     BAD_COMMAND         ; 007H FORMAT DRIVE
7035 00005321 [B1540000]          <1>       dd     BAD_COMMAND         ; 008H RETURN PARAMETERS
7036 00005325 [48560000]          <1>       dd     INIT_DRV            ; 009H
7037 00005329 [A7560000]          <1>       dd     RD_LONG             ; 00AH
7038 0000532D [B0560000]          <1>       dd     WR_LONG             ; 00BH
7039 00005331 [B9560000]          <1>       dd     DISK_SEEK           ; 00CH
7040 00005335 [BB540000]          <1>       dd     DISK_RESET          ; 00DH
7041 00005339 [B1540000]          <1>       dd     BAD_COMMAND         ; 00EH READ BUFFER
7042 0000533D [B1540000]          <1>       dd     BAD_COMMAND         ; 00FH WRITE BUFFER
7043 00005341 [E1560000]          <1>       dd     TST_RDY             ; 010H
7044 00005345 [05570000]          <1>       dd     HDISK_RECAL         ; 011H
7045 00005349 [B1540000]          <1>       dd     BAD_COMMAND         ; 012H MEMORY DIAGNOSTIC
7046 0000534D [B1540000]          <1>       dd     BAD_COMMAND         ; 013H DRIVE DIAGNOSTIC
7047 00005351 [3B570000]          <1>       dd     CTLR_DIAGNOSTIC     ; 014H CONTROLLER DIAGNOSTIC
7048                              <1>       ; 02/02/2015 (Temporary - Retro UNIX 386 v1 - DISK I/O test)
7049 00005355 [B1540000]          <1>       dd     BAD_COMMAND         ; 015h
7050 00005359 [B1540000]          <1>       dd     BAD_COMMAND         ; 016h
7051 0000535D [B1540000]          <1>       dd     BAD_COMMAND         ; 017h
7052 00005361 [B1540000]          <1>       dd     BAD_COMMAND         ; 018h
7053 00005365 [B1540000]          <1>       dd     BAD_COMMAND         ; 019h
7054 00005369 [B1540000]          <1>       dd     BAD_COMMAND         ; 01Ah
7055 0000536D [3F550000]          <1>       dd     DISK_READ           ; 01Bh ; LBA read
7056 00005371 [48550000]          <1>       dd     DISK_WRITE          ; 01Ch ; LBA write
7057                              <1> D1L     EQU    $ - D1
7058                              <1>
7059                              <1> DISK_IO_CONT:
7060                              <1>       ;;CALL DDS                 ; ESTABLISH SEGMENT
7061 00005375 80FC01              <1>       CMP    AH,01H              ; RETURN STATUS
7062                              <1>       ;;JNZ  short SU0
7063                              <1>       ;;JMP  RETURN_STATUS
7064                              <1>       ;je    RETURN_STATUS
7065                              <1>       ; 11/04/2021
7066 00005378 7505                <1>       jne    short SU0
7067 0000537A E9B3010000          <1>       jmp    RETURN_STATUS
7068                              <1> SU0:
7069 0000537F C605[43810100]00    <1>       MOV    byte [DISK_STATUS1],0 ; RESET THE STATUS INDICATOR
7070                              <1>       ;;PUSH BX                  ; SAVE DATA ADDRESS
7071                              <1>       ;mov   si, bx ;; 14/02/2015
7072 00005386 89DE                <1>       mov    esi, ebx ; 21/02/2015
7073 00005388 8A1D[44810100]      <1>       MOV    BL,[HF_NUM]         ; GET NUMBER OF DRIVES
7074                              <1>       ;; 04/01/2015
7075                              <1>       ;;PUSH AX
7076 0000538E 80E27F              <1>       AND    DL,7FH              ; GET DRIVE AS 0 OR 1
7077                              <1>                                  ; (get drive number as 0 to 3)
7078 00005391 38D3                <1>       CMP    BL,DL
7079                              <1>       ;;JBE  BAD_COMMAND_POP     ; INVALID DRIVE
7080 00005393 0F8618010000        <1>       jbe    BAD_COMMAND ;; 14/02/2015
7081                              <1>         ;
7082                              <1>       ;;03/01/2015
7083 00005399 29DB                <1>       sub    ebx, ebx
7084 0000539B 88D3                <1>       mov    bl, dl
7085                              <1>       ;sub   bh, bh
7086 0000539D 883D[58810100]      <1>       mov    [LBAMode], bh       ; 0
7087                              <1>       ;;test byte [bx+hd0_type], 1 ; LBA ready ?
7088                              <1>       ;test  byte [ebx+hd0_type], 1
7089                              <1>       ;jz    short su1           ; no
7090                              <1>       ;inc   byte [LBAMode]
7091                              <1> ;su1:
7092                              <1>       ; 11/04/2021 (32 bit push/pop)
7093                              <1>       ; 21/02/2015 (32 bit modification)
7094                              <1>       ;04/01/2015
7095 000053A3 50                  <1>       push   Eax ; ***
7096                              <1>       ;PUSH  ES ; **
7097 000053A4 52                  <1>       PUSH   eDX ; *
7098 000053A5 50                  <1>       push   Eax ; ****
7099 000053A6 E873060000          <1>       CALL   GET_VEC             ; GET DISK PARAMETERS
7100                              <1>       ; 02/02/2015
7101                              <1>       ;mov   ax, [ES:BX+16] ; I/O port base address (1F0h, 170h)
7102 000053AB 668B4310            <1>       mov    ax, [ebx+16]
7103 000053AF 66A3[E0680000]      <1>       mov    [HF_PORT], ax
7104                              <1>       ;mov   dx, [ES:BX+18] ; control port address (3F6h, 376h)
7105 000053B5 668B5312            <1>       mov    dx, [ebx+18]
7106 000053B9 668915[E2680000]    <1>       mov    [HF_REG_PORT], dx
7107                              <1>       ;mov   al, [ES:BX+20] ; head register upper nibble (A0h,B0h,E0h,F0h)
7108 000053C0 8A4314              <1>       mov    al, [ebx+20]
7109                              <1>       ; 23/02/2015
7110 000053C3 A840                <1>       test   al, 40h       ; LBA bit (bit 6)
7111 000053C5 7406                <1>       jz     short su1
7112 000053C7 FE05[58810100]      <1>       inc    byte [LBAMode] ; 1
7113                              <1> su1:
7114 000053CD C0E804              <1>       shr    al, 4
7115 000053D0 2401                <1>       and    al, 1
7116 000053D2 A2[E4680000]        <1>       mov    [hf_m_s], al
7117                              <1>       ;
7118                              <1>       ; 03/01/2015
7119                              <1>       ;MOV   AL,byte [ES:BX+8]   ; GET CONTROL BYTE MODIFIER
7120 000053D7 8A4308              <1>       mov    al, [ebx+8]
7121                              <1>       ;MOV   DX,[HF_REG_PORT]    ; Device Control register
7122 000053DA EE                  <1>       OUT    DX,AL               ; SET EXTRA HEAD OPTION
7123                              <1>                                  ; Control Byte:  (= 08h, here)
7124                              <1>                                  ; bit 0 - 0
7125                              <1>                                  ; bit 1 - nIEN (1 = disable irq)
7126                              <1>                                  ; bit 2 - SRST (software RESET)
7127                              <1>                                  ; bit 3 - use extra heads (8 to 15)
7128                              <1>                                  ;         -always set to 1-
7129                              <1>                                  ; (bits 3 to 7 are reserved
7130                              <1>                                  ;          for ATA devices)
7131 000053DB 8A25[45810100]      <1>       MOV    AH,[CONTROL_BYTE]   ; SET EXTRA HEAD OPTION IN
7132 000053E1 80E4C0              <1>       AND    AH,0C0H             ; CONTROL BYTE
7133 000053E4 08C4                <1>       OR     AH,AL
7134 000053E6 8825[45810100]      <1>       MOV    [CONTROL_BYTE],AH
```

```
7135                            <1>       ; 11/04/2021 (32 bit push/pop)
7136                            <1>       ; 04/01/2015
7137 000053EC 58                <1>       pop    Eax ; ****
7138 000053ED 5A                <1>       pop    Edx ; * ;; 14/02/2015
7139 000053EE 20E4              <1>       and    ah, ah ; Reset function ?
7140 000053F0 7506              <1>       jnz    short su2
7141                            <1>       ;;pop  dx ; * ;; 14/02/2015
7142                            <1>       ;pop   es ; **
7143 000053F2 58                <1>       pop    Eax ; ***
7144                            <1>       ;;pop  bx
7145 000053F3 E9C3000000        <1>         jmp     DISK_RESET
7146                            <1> su2:
7147 000053F8 803D[58810100]00  <1>       cmp    byte [LBAMode], 0
7148 000053FF 7660              <1>       jna    short su3
7149                            <1>       ;
7150                            <1>       ; 02/02/2015 (LBA read/write function calls)
7151 00005401 80FC1B            <1>       cmp    ah, 1Bh
7152 00005404 720B              <1>       jb     short lbarw1
7153 00005406 80FC1C            <1>       cmp    ah, 1Ch
7154 00005409 775B              <1>       ja     short invldfnc
7155                            <1>       ;;pop  dx ; * ; 14/02/2015
7156                            <1>       ;mov   ax, cx ; Lower word of LBA address (bits 0-15)
7157 0000540B 89C8              <1>       mov    eax, ecx ; LBA address (21/02/2015)
7158                            <1>       ;; 14/02/2015
7159 0000540D 88D1              <1>       mov    cl, dl ; 14/02/2015
7160                            <1>       ;;mov  dx, bx
7161                            <1>       ;mov   dx, si ; higher word of LBA address (bits 16-23)
7162                            <1>       ;;mov  bx, di
7163                            <1>       ;mov   si, di ; Buffer offset
7164 0000540F EB30              <1>       jmp    short lbarw2
7165                            <1> lbarw1:
7166                            <1>       ; convert CHS to LBA
7167                            <1>       ;
7168                            <1>       ; LBA calculation - AWARD BIOS - 1999 - AHDSK.ASM
7169                            <1>       ; LBA = "# of Heads" * Sectors/Track * Cylinder + Head * Sectors/Track
7170                            <1>       ;      + Sector - 1
7171                            <1>       ; 11/04/2021 (32 bit push/pop)
7172 00005411 52                <1>       push   Edx ; * ;; 14/02/2015
7173                            <1>       ;xor   dh, dh
7174 00005412 31D2              <1>       xor    edx, edx
7175                            <1>       ;mov   dl, [ES:BX+14]      ; sectors per track (logical)
7176 00005414 8A530E            <1>       mov    dl, [ebx+14]
7177                            <1>       ;xor   ah, ah
7178 00005417 31C0              <1>       xor    eax, eax
7179                            <1>       ;mov   al, [ES:BX+2]; heads (logical)
7180 00005419 8A4302            <1>       mov    al, [ebx+2]
7181 0000541C FEC8              <1>       dec    al
7182 0000541E 6640              <1>       inc    ax           ; 0 =  256
7183 00005420 66F7E2            <1>       mul    dx
7184                            <1>              ; AX = # of Heads" * Sectors/Track
7185 00005423 6689CA            <1>       mov    dx, cx
7186                            <1>       ;and   cx, 3Fh       ; sector  (1 to 63)
7187 00005426 83E13F            <1>       and    ecx, 3fh
7188 00005429 86D6              <1>       xchg   dl, dh
7189 0000542B C0EE06            <1>       shr    dh, 6
7190                            <1>              ; DX = cylinder (0 to 1023)
7191                            <1>       ;mul   dx
7192                            <1>              ; DX:AX = # of Heads" * Sectors/Track * Cylinder
7193 0000542E F7E2              <1>       mul    edx
7194 00005430 FEC9              <1>       dec    cl  ; sector - 1
7195                            <1>       ;add   ax, cx
7196                            <1>       ;adc   dx, 0
7197                            <1>              ; DX:AX = # of Heads" * Sectors/Track * Cylinder + Sector -1
7198 00005432 01C8              <1>       add    eax, ecx
7199                            <1>       ; 11/04/2021 (32 bit push/pop)
7200 00005434 59                <1>       pop    Ecx ; * ; ch = head, cl = drive number (zero based)
7201                            <1>       ;push  dx
7202                            <1>       ;push  ax
7203 00005435 50                <1>       push   eax
7204                            <1>       ;mov   al, [ES:BX+14]      ; sectors per track (logical)
7205 00005436 8A430E            <1>       mov    al, [ebx+14]
7206 00005439 F6E5              <1>       mul    ch
7207                            <1>              ; AX = Head * Sectors/Track
7208 0000543B 0FB7C0            <1>       movzx  eax, ax ; 09/12/2017
7209                            <1>       ;pop   dx
7210 0000543E 5A                <1>       pop    edx
7211                            <1>       ;add   ax, dx
7212                            <1>       ;pop   dx
7213                            <1>       ;adc   dx, 0 ; add carry bit
7214 0000543F 01D0              <1>       add    eax, edx
7215                            <1> lbarw2:
7216 00005441 29D2              <1>       sub    edx, edx ; 21/02/2015
7217 00005443 88CA              <1>       mov    dl, cl ; 21/02/2015
7218 00005445 C645F800          <1>       mov   byte [CMD_BLOCK], 0 ; Features Register
7219                            <1>                              ; NOTE: Features register (1F1h, 171h)
7220                            <1>                              ; is not used for ATA device R/W functions.
7221                            <1>                              ; It is old/obsolete 'write precompensation'
7222                            <1>                              ; register and error register
7223                            <1>                              ; for old ATA/IDE devices.
7224                            <1>       ; 18/01/2014
7225                            <1>       ;mov   ch, [hf_m_s] ; Drive 0 (master) or 1 (slave)
7226 00005449 8A0D[E4680000]    <1>       mov    cl, [hf_m_s]
7227                            <1>       ;shl   ch, 4        ; bit 4 (drive bit)
7228                            <1>       ;or    ch, 0E0h     ; bit 5 = 1
7229                            <1>                           ; bit 6 = 1 = LBA mode
7230                            <1>                           ; bit 7 = 1
7231 0000544F 80C90E            <1>       or     cl, 0Eh ; 1110b
7232                            <1>       ;and   dh, 0Fh             ; LBA byte 4 (bits 24 to 27)
7233 00005452 25FFFFFF0F        <1>       and    eax, 0FFFFFFFh
7234 00005457 C1E11C            <1>       shl    ecx, 28 ; 21/02/2015
7235                            <1>       ;or    dh, ch
7236 0000545A 09C8              <1>       or     eax, ecx
7237                            <1>       ;;mov  [CMD_BLOCK+2], al ; LBA byte 1 (bits 0 to 7)
7238                            <1>                              ; (Sector Number Register)
7239                            <1>       ;;mov  [CMD_BLOCK+3], ah ; LBA byte 2 (bits 8 to 15)
```

```
                                <1>                             ; (Cylinder Low Register)
                                <1>       ;mov   [CMD_BLOCK+2], ax ; LBA byte 1, 2
                                <1>       ;mov   [CMD_BLOCK+4], dl ; LBA byte 3 (bits 16 to 23)
                                <1>                             ; (Cylinder High Register)
                                <1>       ;;mov  [CMD_BLOCK+5], dh ; LBA byte 4 (bits 24 to 27)
                                <1>                             ; (Drive/Head Register)
                                <1>
                                <1>       ;mov   [CMD_BLOCK+4], dx ; LBA byte 4, LBA & DEV select bits
0000545C 8945FA                 <1>       mov    [CMD_BLOCK+2], eax ; 21/02/2015
                                <1>       ;14/02/2015
                                <1>       ;mov   dl, cl ; Drive number (INIT_DRV)
0000545F EB37                   <1>       jmp    short su4
                                <1> su3:
                                <1>       ; 02/02/2015
                                <1>       ; (Temporary functions 1Bh & 1Ch are not valid for CHS mode)
00005461 80FC14                 <1>       cmp    ah, 14h
00005464 7603                   <1>       jna    short chsfnc
                                <1> invldfnc:
                                <1>         ; 14/02/2015
                                <1>       ;pop   es ; **
                                <1>       ;pop    ax ; ***
                                <1>       ; 11/04/2021
00005466 58                     <1>       pop    eax ; ***
                                <1>       ;jmp   short BAD_COMMAND_POP
00005467 EB48                   <1>       jmp   short BAD_COMMAND
                                <1> chsfnc:
                                <1>       ;MOV   AX,[ES:BX+5]       ; GET WRITE PRE-COMPENSATION CYLINDER
00005469 668B4305               <1>       mov    ax, [ebx+5]
0000546D 66C1E802               <1>       SHR    AX,2
00005471 8845F8                 <1>       MOV    [CMD_BLOCK],AL
                                <1>       ;;MOV  AL,[ES:BX+8]       ; GET CONTROL BYTE MODIFIER
                                <1>       ;;PUSH DX
                                <1>       ;;MOV  DX,[HF_REG_PORT]
                                <1>       ;;OUT  DX,AL              ; SET EXTRA HEAD OPTION
                                <1>       ;;POP  DX ; *
                                <1>       ;;POP  ES ; **
                                <1>       ;;MOV  AH,[CONTROL_BYTE]  ; SET EXTRA HEAD OPTION IN
                                <1>       ;;AND  AH,0C0H            ; CONTROL BYTE
                                <1>       ;;OR   AH,AL
                                <1>       ;;MOV  [CONTROL_BYTE],AH
                                <1>       ;
00005474 88C8                   <1>       MOV    AL,CL              ; GET SECTOR NUMBER
00005476 243F                   <1>       AND    AL,3FH
00005478 8845FA                 <1>       MOV    [CMD_BLOCK+2],AL
0000547B 886DFB                 <1>       MOV    [CMD_BLOCK+3],CH   ; GET CYLINDER NUMBER
0000547E 88C8                   <1>       MOV    AL,CL
00005480 C0E806                 <1>       SHR    AL,6
00005483 8845FC                 <1>       MOV    [CMD_BLOCK+4],AL   ; CYLINDER HIGH ORDER 2 BITS
                                <1>       ;;05/01/2015
                                <1>       ;;MOV  AL,DL              ; DRIVE NUMBER
00005486 A0[E4680000]           <1>       mov    al, [hf_m_s]
0000548B C0E004                 <1>       SHL    AL,4
0000548E 80E60F                 <1>       AND    DH,0FH             ; HEAD NUMBER
00005491 08F0                   <1>       OR     AL,DH
                                <1>       ;OR    AL,80H or 20H
00005493 0CA0                   <1>       OR     AL,80h+20h         ; ECC AND 512 BYTE SECTORS
00005495 8845FD                 <1>       MOV    [CMD_BLOCK+5],AL   ; ECC/SIZE/DRIVE/HEAD
                                <1> su4:
                                <1>       ;POP   ES ; **
                                <1>       ;; 14/02/2015
                                <1>       ;;POP   AX
                                <1>       ;;MOV   [CMD_BLOCK+1],AL  ; SECTOR COUNT
                                <1>       ;;PUSH  AX
                                <1>       ;;MOV   AL,AH             ; GET INTO LOW BYTE
                                <1>       ;;XOR   AH,AH             ; ZERO HIGH BYTE
                                <1>       ;;SAL   AX,1              ; *2 FOR TABLE LOOKUP
                                <1>       ;pop   ax ; ***
                                <1>         ; 11/04/2021
00005498 58                     <1>       pop    eax ; ***
00005499 8845F9                 <1>       mov   [CMD_BLOCK+1], al
0000549C 29DB                   <1>       sub    ebx, ebx
0000549E 88E3                   <1>       mov   bl, ah
                                <1>       ;xor   bh, bh
                                <1>       ;sal   bx, 1
000054A0 66C1E302               <1>       sal    bx, 2  ; 32 bit offset (21/02/2015)
                                <1>       ;;MOV  SI,AX              ; PUT INTO SI FOR BRANCH
                                <1>       ;;CMP  AX,D1L             ; TEST WITHIN RANGE
                                <1>       ;;JNB  short BAD_COMMAND_POP
                                <1>       ;cmp   bx, D1L
000054A4 83FB74                 <1>       cmp    ebx, D1L
000054A7 7308                   <1>       jnb    short BAD_COMMAND
                                <1>       ;xchg  bx, si
000054A9 87DE                   <1>       xchg   ebx, esi
                                <1>       ;;;POP AX                 ; RESTORE AX
                                <1>       ;;;POP BX                 ; AND DATA ADDRESS
                                <1>
                                <1>       ;;PUSH CX
                                <1>       ;;PUSH AX                 ; ADJUST ES:BX
                                <1>       ;MOV   CX,BX              ; GET 3 HIGH ORDER NIBBLES OF BX
                                <1>       ;SHR   CX,4
                                <1>       ;MOV   AX,ES
                                <1>       ;ADD   AX,CX
                                <1>       ;MOV   ES,AX
                                <1>       ;AND   BX,000FH           ; ES:BX CHANGED TO ES:000X
                                <1>       ;;POP  AX
                                <1>       ;;POP  CX
                                <1>       ;;JMP  word [CS:SI+D1]
                                <1>       ;jmp   word [SI+D1]
000054AB FFA6[01530000]         <1>       jmp    dword [esi+D1]
                                <1> ;;BAD_COMMAND_POP:
                                <1> ;;    POP    AX
                                <1> ;;    POP    BX
                                <1> BAD_COMMAND:
000054B1 C605[43810100]01       <1>       MOV    byte [DISK_STATUS1],BAD_CMD ; COMMAND ERROR
000054B8 B000                   <1>       MOV    AL,0
```

```
7345 000054BA C3                  <1>       RETn
7346                              <1>
7347                              <1> ;--------------------------------------
7348                              <1> ;     RESET THE DISK SYSTEM  (AH=00H) :
7349                              <1> ;--------------------------------------
7350                              <1>
7351                              <1> ; 18-1-2015 : one controller reset (not other one)
7352                              <1>
7353                              <1> DISK_RESET:
7354 000054BB FA                  <1>       CLI
7355 000054BC E4A1                <1>       IN     AL,INTB01          ; GET THE MASK REGISTER
7356                              <1>       ;JMP   $+2
7357                              <1>       IODELAY
2990 000054BE EB00                <2>  jmp short $+2
2991 000054C0 EB00                <2>  jmp short $+2
7358                              <1>       ;AND   AL,0BFH             ; ENABLE FIXED DISK INTERRUPT
7359 000054C2 243F                <1>       and    al,3Fh              ; 22/12/2014 (IRQ 14 & IRQ 15)
7360 000054C4 E6A1                <1>       OUT    INTB01,AL
7361 000054C6 FB                  <1>       STI                        ; START INTERRUPTS
7362                              <1>       ; 14/02/2015
7363 000054C7 6689D7              <1>       mov    di, dx
7364                              <1>       ; 04/01/2015
7365                              <1>       ;xor   di,di
7366                              <1> drst0:
7367 000054CA B004                <1>       MOV    AL,04H  ; bit 2 - SRST
7368                              <1>       ;MOV   DX,HF_REG_PORT
7369 000054CC 668B15[E2680000]    <1>       MOV    DX,[HF_REG_PORT]
7370 000054D3 EE                  <1>       OUT    DX,AL               ; RESET
7371                              <1> ;     MOV    CX,10               ; DELAY COUNT
7372                              <1> ;DRD: DEC    CX
7373                              <1> ;     JNZ    short DRD           ; WAIT 4.8 MICRO-SEC
7374                              <1>       ;mov   cx,2                ; wait for 30 micro seconds
7375 000054D4 B902000000          <1>       mov    ecx, 2 ; 21/02/2015
7376 000054D9 E8CACEFFFF          <1>       call  WAITF                ; (Award Bios 1999 - WAIT_REFRESH,
7377                              <1>                                  ; 40 micro seconds)
7378 000054DE A0[45810100]        <1>       mov    al,[CONTROL_BYTE]
7379 000054E3 240F                <1>       AND    AL,0FH              ; SET HEAD OPTION
7380 000054E5 EE                  <1>       OUT    DX,AL               ; TURN RESET OFF
7381 000054E6 E82C040000          <1>       CALL   NOT_BUSY
7382 000054EB 7515                <1>       JNZ    short DRERR         ; TIME OUT ON RESET
7383 000054ED 668B15[E0680000]    <1>       MOV    DX,[HF_PORT]
7384 000054F4 FEC2                <1>       inc    dl  ; HF_PORT+1
7385                              <1>       ; 02/01/2015 - Award BIOS 1999 - AHDSK.ASM
7386                              <1>       ;mov   cl, 10
7387 000054F6 B90A000000          <1>       mov    ecx, 10 ; 21/02/2015
7388                              <1> drst1:
7389 000054FB EC                  <1>       IN     AL,DX               ; GET RESET STATUS
7390 000054FC 3C01                <1>       CMP    AL,1
7391                              <1>       ; 04/01/2015
7392 000054FE 740A                <1>       jz     short drst2
7393                              <1>       ;JNZ   short DRERR         ; BAD RESET STATUS
7394                              <1>              ; Drive/Head Register - bit 4
7395 00005500 E2F9                <1>       loop   drst1
7396                              <1> DRERR:
7397 00005502 C605[43810100]05    <1>       MOV    byte [DISK_STATUS1],BAD_RESET ; CARD FAILED
7398 00005509 C3                  <1>       RETn
7399                              <1> drst2:
7400                              <1>       ; 14/02/2015
7401 0000550A 6689FA              <1>       mov    dx,di
7402                              <1> ;drst3:
7403                              <1> ;     ; 05/01/2015
7404                              <1> ;     shl    di,1
7405                              <1> ;     ; 04/01/2015
7406                              <1> ;     mov    ax,[di+hd_cports]
7407                              <1> ;     cmp    ax,[HF_REG_PORT]
7408                              <1> ;     je     short drst4
7409                              <1> ;     mov    [HF_REG_PORT], ax
7410                              <1> ;     ; 03/01/2015
7411                              <1> ;     mov    ax,[di+hd_ports]
7412                              <1> ;     mov   [HF_PORT], ax
7413                              <1> ;     ; 05/01/2014
7414                              <1> ;     shr    di,1
7415                              <1> ;     ; 04/01/2015
7416                              <1> ;     jmp    short drst0  ; reset other controller
7417                              <1> ;drst4:
7418                              <1> ;     ; 05/01/2015
7419                              <1> ;     shr    di,1
7420                              <1> ;     mov    al,[di+hd_dregs]
7421                              <1> ;     and    al,10h ; bit 4 only
7422                              <1> ;     shr    al,4 ; bit 4  -> bit 0
7423                              <1> ;     mov    [hf_m_s], al ; (0 = master, 1 = slave)
7424                              <1>       ;
7425 0000550D A0[E4680000]        <1>       mov    al, [hf_m_s] ; 18/01/2015
7426 00005512 A801                <1>       test   al,1
7427                              <1> ;     jnz    short drst6
7428 00005514 7516                <1>       jnz   short drst4
7429 00005516 8065FDEF            <1>       AND   byte [CMD_BLOCK+5],0EFH ; SET TO DRIVE 0
7430                              <1> ;drst5:
7431                              <1> drst3:
7432 0000551A E829010000          <1>       CALL   INIT_DRV            ; SET MAX HEADS
7433                              <1>       ;mov   dx,di
7434 0000551F E8E1010000          <1>       CALL   HDISK_RECAL         ; RECAL TO RESET SEEK SPEED
7435                              <1>       ; 04/01/2014
7436                              <1> ;     inc    di
7437                              <1> ;     mov    dx,di
7438                              <1> ;     cmp    dl,[HF_NUM]
7439                              <1> ;     jb     short drst3
7440                              <1> ;DRE:
7441 00005524 C605[43810100]00    <1>       MOV    byte [DISK_STATUS1],0     ; IGNORE ANY SET UP ERRORS
7442 0000552B C3                  <1>       RETn
7443                              <1> ;drst6:
7444                              <1> drst4:       ; Drive/Head Register - bit 4
7445 0000552C 804DFD10            <1>       OR     byte [CMD_BLOCK+5],010H ; SET TO DRIVE 1
7446                              <1>      ;jmp    short drst5
7447 00005530 EBE8                <1>       jmp    short drst3
```

```
7448                                    <1>
7449                                    <1> ;--------------------------------------
7450                                    <1> ;     DISK STATUS ROUTINE  (AH = 01H) :
7451                                    <1> ;--------------------------------------
7452                                    <1>
7453                                    <1> RETURN_STATUS:
7454 00005532 A0[43810100]             <1>       MOV    AL,[DISK_STATUS1]   ; OBTAIN PREVIOUS STATUS
7455 00005537 C605[43810100]00         <1>       MOV   byte [DISK_STATUS1],0 ; RESET STATUS
7456 0000553E C3                       <1>       RETn
7457                                    <1>
7458                                    <1> ;--------------------------------------
7459                                    <1> ;     DISK READ ROUTINE    (AH = 02H) :
7460                                    <1> ;--------------------------------------
7461                                    <1>
7462                                    <1> DISK_READ:
7463 0000553F C645FE20                 <1>       MOV    byte [CMD_BLOCK+6],READ_CMD
7464 00005543 E948020000               <1>       JMP   COMMANDI
7465                                    <1>
7466                                    <1> ;--------------------------------------
7467                                    <1> ;     DISK WRITE ROUTINE   (AH = 03H) :
7468                                    <1> ;--------------------------------------
7469                                    <1>
7470                                    <1> DISK_WRITE:
7471 00005548 C645FE30                 <1>       MOV    byte [CMD_BLOCK+6],WRITE_CMD
7472 0000554C E99A020000               <1>       JMP   COMMANDO
7473                                    <1>
7474                                    <1> ;--------------------------------------
7475                                    <1> ;     DISK VERIFY       (AH = 04H) :
7476                                    <1> ;--------------------------------------
7477                                    <1>
7478                                    <1> DISK_VERF:
7479 00005551 C645FE40                 <1>       MOV    byte [CMD_BLOCK+6],VERIFY_CMD
7480 00005555 E808030000               <1>       CALL   COMMAND
7481 0000555A 750C                     <1>       JNZ    short VERF_EXIT     ; CONTROLLER STILL BUSY
7482 0000555C E87A030000               <1>       CALL   _WAIT               ; (Original: CALL WAIT)
7483 00005561 7505                     <1>       JNZ    short VERF_EXIT     ; TIME OUT
7484 00005563 E807040000               <1>       CALL   CHECK_STATUS
7485                                    <1> VERF_EXIT:
7486 00005568 C3                       <1>       RETn
7487                                    <1>
7488                                    <1> ;--------------------------------------
7489                                    <1> ;     FORMATTING        (AH = 05H) :
7490                                    <1> ;--------------------------------------
7491                                    <1>
7492                                    <1> FMT_TRK:                          ; FORMAT TRACK  (AH = 005H)
7493 00005569 C645FE50                 <1>       MOV    byte [CMD_BLOCK+6],FMTTRK_CMD
7494                                    <1>       ;PUSH  ES
7495                                    <1>       ;PUSH  BX
7496 0000556D 53                       <1>       push   ebx
7497 0000556E E8AB040000               <1>       CALL   GET_VEC             ; GET DISK PARAMETERS ADDRESS
7498                                    <1>       ;MOV   AL,[ES:BX+14]       ; GET SECTORS/TRACK
7499 00005573 8A430E                   <1>       mov    al, [ebx+14]
7500 00005576 8845F9                   <1>       MOV    [CMD_BLOCK+1],AL    ; SET SECTOR COUNT IN COMMAND
7501 00005579 5B                       <1>       pop    ebx
7502                                    <1>       ;POP   BX
7503                                    <1>       ;POP   ES
7504 0000557A E973020000               <1>       JMP   CMD_OF               ; GO EXECUTE THE COMMAND
7505                                    <1>
7506                                    <1> ;--------------------------------------
7507                                    <1> ;     READ DASD TYPE           (AH = 15H) :
7508                                    <1> ;--------------------------------------
7509                                    <1>
7510                                    <1> READ_DASD_TYPE:
7511                                    <1> READ_D_T:                         ; GET DRIVE PARAMETERS
7512 0000557F 1E                       <1>       PUSH   DS                   ; SAVE REGISTERS
7513                                    <1>       ;;PUSH ES
7514                                    <1>       ; 18/04/2021
7515                                    <1>       ;PUSH  eBX
7516                                    <1>       ;;CALL DDS                  ; ESTABLISH ADDRESSING
7517                                    <1>       ;;push cs
7518                                    <1>       ;;pop  ds
7519                                    <1>         ; 18/04/2021
7520                                    <1>       ;mov   bx, KDATA
7521                                    <1>       ;mov   ds, bx
7522                                    <1>       ;;mov  es, bx
7523                                    <1>       ;MOV   byte [DISK_STATUS1],0
7524                                    <1>       ;MOV   BL,[HF_NUM]          ; GET NUMBER OF DRIVES
7525                                    <1>       ;AND   DL,7FH               ; GET DRIVE NUMBER
7526                                    <1>       ;CMP   BL,DL
7527                                    <1>       ;JBE   short RDT_NOT_PRESENT ; RETURN DRIVE NOT PRESENT
7528 00005580 66B81000                 <1>       mov    ax, KDATA
7529 00005584 8ED8                     <1>       mov    ds, ax
7530 00005586 C605[43810100]00         <1>       mov    byte [DISK_STATUS1], 0
7531 0000558D 8A0D[44810100]           <1>       mov    cl, [HF_NUM]
7532 00005593 80E27F                   <1>       and    dl, 7Fh
7533 00005596 38D1                     <1>       cmp    cl, dl
7534 00005598 7622                     <1>       jbe    short RDT_NOT_PRESENT
7535                                    <1>
7536                                    <1>       ; 18/04/2021 - TRDOS 386 v2.0.4
7537                                    <1>
7538                                    <1>       ;CALL  GET_VEC              ; GET DISK PARAMETER ADDRESS
7539                                    <1>       ;
7540                                    <1>       ;;MOV  AL,[ES:BX+2]         ; HEADS
7541                                    <1>       ;mov   al, [ebx+2]  ; heads (logical)
7542                                    <1>       ;;;MOV CL,[ES:BX+14]
7543                                    <1>       ;;mov  cl, [ebx+14]
7544                                    <1>       ;; 17/04/2021
7545                                    <1>       ;mov   ah, [ebx+14] ; sectors per track (logical)
7546                                    <1>       ;;IMUL CL             ; * NUMBER OF SECTORS
7547                                    <1>       ;;MOV  CX,[ES:BX]     ; MAX NUMBER OF CYLINDERS
7548                                    <1>       ;mov   cx, [ebx]    ; cylinders (logical)
7549                                    <1>       ;; 02/01/2015
7550                                    <1>       ;; ** leave the last cylinder as reserved for diagnostics **
7551                                    <1>       ;; (Also in Award BIOS - 1999, AHDSK.ASM, FUN15 -> sub ax, 1)
7552                                    <1>       ;DEC   CX                   ; LEAVE ONE FOR DIAGNOSTICS
```

```
7553                            <1>       ;;IMUL CX                 ; NUMBER OF SECTORS
7554                            <1>       ;; 17/04/2021
7555                            <1>       ;mul   ah
7556                            <1>       ;; ax = spt*heads
7557                            <1>       ;mul   cx
7558                            <1>       ;; dx:ax = number of sectors
7559                            <1>       ;
7560                            <1>       ;MOV   CX,DX              ; HIGH ORDER HALF
7561                            <1>       ;MOV   DX,AX              ; LOW ORDER HALF
7562                            <1>
7563                            <1>       ; 18/04/2021
7564 0000559A B102             <1>       mov    cl, 2
7565 0000559C 00CA             <1>       add    dl, cl ; hd0 = 2
7566 0000559E D2E2             <1>       shl    dl, cl ; * 4
7567 000055A0 0FB6D2           <1>       movzx  edx, dl
7568 000055A3 8B82[1E690000]   <1>       mov    eax, [edx+drv.size]
7569 000055A9 6689C2           <1>       mov    dx, ax
7570 000055AC C1E810           <1>       shr    eax, 16
7571 000055AF 89C1             <1>       mov    ecx, eax
7572                            <1>
7573                            <1>       ;SUB   AX,AX
7574 000055B1 28C0             <1>       sub    al, al
7575 000055B3 B403             <1>       MOV    AH,03H             ; INDICATE FIXED DISK
7576                            <1> RDT2:
7577                            <1>       ; 18/04/2021
7578                            <1>       ;POP   eBX                ; RESTORE REGISTERS
7579                            <1>       ;;POP  ES
7580 000055B5 1F               <1>       POP    DS
7581                            <1>       ; (*) CLC                 ; CLEAR CARRY
7582                            <1>       ;RETf  2
7583                            <1>       ; (*) 29/05/2016
7584                            <1>       ; (*) retf 4
7585 000055B6 80642408FE       <1>       and    byte [esp+8], 0FEh ; clear carry bit of eflags register
7586 000055BB CF               <1>       iretd
7587                            <1>
7588                            <1> RDT_NOT_PRESENT:
7589                            <1>       ;SUB   AX,AX              ; DRIVE NOT PRESENT RETURN
7590                            <1>       ; 18/04/2021
7591 000055BC 29C0             <1>       sub    eax, eax
7592                            <1>       ;MOV   CX,AX              ; ZERO BLOCK COUNT
7593                            <1>       ;MOV   DX,AX
7594 000055BE 89C1             <1>       mov    ecx, eax
7595 000055C0 89C2             <1>       mov    edx, eax
7596 000055C2 EBF1             <1>       JMP    short RDT2
7597                            <1>
7598                            <1> ; 28/05/2016
7599                            <1> ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
7600                            <1>
7601                            <1> ;----------------------------------------
7602                            <1> ;     GET PARAMETERS          (AH = 08H) :
7603                            <1> ;----------------------------------------
7604                            <1>
7605                            <1> GET_PARM_N:
7606                            <1>       ; ebx = user's buffer address for parameters table
7607                            <1> ;GET_PARM:                        ; GET DRIVE PARAMETERS
7608 000055C4 1E               <1>       PUSH   DS                 ; SAVE REGISTERS
7609 000055C5 06               <1>       PUSH   ES
7610 000055C6 53               <1>       PUSH   eBX
7611                            <1>       ;MOV   AX,ABS0            ; ESTABLISH ADDRESSING
7612                            <1>       ;MOV   DS,AX
7613                            <1>       ;TEST  DL,1               ; CHECK FOR DRIVE 1
7614                            <1>       ;JZ    short G0
7615                            <1>       ;LES   BX,@HF1_TBL_VEC
7616                            <1>       ;JMP   SHORT G1
7617                            <1> ;G0:  LES    BX,@HF_TBL_VEC
7618                            <1> ;G1:
7619                            <1>       ;CALL  DDS                ; ESTABLISH SEGMENT
7620                            <1>       ; 22/12/2014
7621                            <1>       ;push  cs
7622                            <1>       ;pop   ds
7623 000055C7 66BB1000         <1>       mov    bx, KDATA
7624 000055CB 8EDB             <1>       mov    ds, bx
7625 000055CD 8EC3             <1>       mov    es, bx ; 27/05/2016
7626                            <1>       ;
7627                            <1>       ; 18/04/2021
7628 000055CF 29C9             <1>       sub    ecx, ecx
7629                            <1>       ;
7630 000055D1 80EA80           <1>       SUB    DL,80H
7631 000055D4 80FA04           <1>       CMP    DL,MAX_FILE        ; TEST WITHIN RANGE
7632 000055D7 7361             <1>       JAE    short G4
7633                            <1>       ;
7634                            <1>       ; 21/02/2015
7635 000055D9 31DB             <1>       xor    ebx, ebx
7636                            <1>       ; 18/04/2021
7637                            <1>       ;sub   ecx, ecx
7638                            <1>       ; 22/12/2014
7639 000055DB 88D3             <1>       mov    bl, dl
7640                            <1>       ;xor   bh, bh
7641 000055DD C0E302           <1>       shl    bl, 2              ; convert index to offset
7642                            <1>       ;add   bx, HF_TBL_VEC
7643 000055E0 81C3[48810100]   <1>       add    ebx, HF_TBL_VEC
7644                            <1>       ;mov   ax, [bx+2]
7645                            <1>       ;mov   es, ax             ; dpt segment
7646                            <1>       ;mov   bx, [bx]           ; dpt offset
7647 000055E6 8B1B             <1>       mov    ebx, [ebx] ; 32 bit offset
7648                            <1>       ; 18/04/2021
7649 000055E8 29D2             <1>       sub    edx, edx
7650 000055EA 8815[43810100]   <1>       mov    [DISK_STATUS1], dl ; 0
7651                            <1>
7652                            <1>       ;MOV   byte [DISK_STATUS1],0
7653                            <1>         ;MOV AX,[ES:BX]         ; MAX NUMBER OF CYLINDERS
7654 000055F0 668B03           <1>       mov    ax, [ebx]
7655                            <1>       ;;SUB  AX,2               ; ADJUST FOR 0-N
7656 000055F3 6648             <1>       dec    ax                 ; max. cylinder number
7657 000055F5 88C5             <1>       MOV    CH,AL
```

```
7658 000055F7 66250003          <1>       AND    AX,0300H           ; HIGH TWO BITS OF CYLINDER
7659 000055FB 66D1E8            <1>       SHR    AX,1
7660 000055FE 66D1E8            <1>       SHR    AX,1
7661                            <1>       ;OR    AL,[ES:BX+14]      ; SECTORS
7662 00005601 0A430E            <1>       or     al, [ebx+14]
7663 00005604 88C1              <1>       MOV    CL,AL
7664                            <1>       ;MOV   DH,[ES:BX+2]       ; HEADS
7665 00005606 8A7302            <1>       mov    dh, [ebx+2]
7666 00005609 FECE              <1>       DEC    DH                 ; 0-N RANGE
7667 0000560B 8A15[44810100]    <1>       MOV    DL,[HF_NUM]        ; DRIVE COUNT
7668                            <1>       ;SUB   AX,AX
7669                            <1>       ; 18/04/2021
7670 00005611 29C0              <1>       sub    eax, eax
7671                            <1>
7672                            <1>       ;27/12/2014
7673                            <1>       ;mov   di, bx             ; HDPT offset
7674                            <1>
7675                            <1>       ; 27/05/2016
7676                            <1>       ; return fixed disk parameters table to user
7677                            <1>       ; in user's buffer, which is pointed by EBX
7678                            <1>       ;
7679 00005613 873C24            <1>       xchg   edi, [esp]         ; ebx (input)-> edi, edi -> [esp]
7680 00005616 56                <1>       push   esi
7681 00005617 89DE              <1>       mov    esi, ebx           ; hard disk parameter table (32 bytes)
7682 00005619 89FB              <1>       mov    ebx, edi           ; ebx = user's buffer address
7683 0000561B 51                <1>       push   ecx
7684 0000561C 50                <1>       push   eax
7685 0000561D B920000000        <1>       mov    ecx, 32 ; 32 bytes
7686 00005622 E81BBF0000        <1>       call   transfer_to_user_buffer ; trdosk6.s (16/05/2016)
7687 00005627 58                <1>       pop    eax
7688 00005628 59                <1>       pop    ecx
7689 00005629 5E                <1>       pop    esi
7690 0000562A 5F                <1>       pop    edi
7691 0000562B 730A              <1>       jnc    short G5
7692                            <1>       ; 29/05/2016 (*)
7693 0000562D B8FF000000        <1>       mov    eax, 0FFh ; unknown error !
7694                            <1> _G6:
7695 00005632 804C241001        <1>       or     byte [esp+16], 1 ; set carry bit of eflags register
7696                            <1> G5:
7697                            <1>       ; 27/05/2016
7698                            <1>       ;POP   eBX                ; RESTORE REGISTERS
7699 00005637 07                <1>       POP    ES
7700 00005638 1F                <1>       POP    DS
7701                            <1>       ;RETf  2
7702                            <1>       ; (*) 29/05/2016
7703                            <1>       ; (*) retf 4
7704                            <1>       ; (*) or byte [esp+8], 1 ; set carry bit of eflags register
7705 00005639 CF                <1>       iretd
7706                            <1> G4:
7707                            <1>       ;MOV   byte [DISK_STATUS1],INIT_FAIL
7708                            <1>       ;                         ; OPERATION FAILED
7709                            <1>       ;MOV   AH,INIT_FAIL
7710                            <1>       ;SUB   AL,AL
7711                            <1>       ;SUB   DX,DX
7712                            <1>       ;SUB   CX,CX
7713                            <1>       ; 18/04/2021
7714 0000563A 29C0              <1>       sub    eax, eax
7715 0000563C B407              <1>       mov    ah, INIT_FAIL
7716 0000563E 8825[43810100]    <1>       mov    [DISK_STATUS1], ah  ; OPERATION FAILED
7717 00005644 29D2              <1>       sub    edx, edx
7718                            <1>       ;sub   ecx, ecx
7719                            <1>
7720                            <1>       ; 29/05/2016 (*)
7721                            <1>       ;STC                      ; SET ERROR FLAG
7722                            <1>       ;JMP   short G5
7723 00005646 EBEA              <1>       jmp    short _G6
7724                            <1>
7725                            <1> ;--------------------------------------
7726                            <1> ;     INITIALIZE DRIVE     (AH = 09H) :
7727                            <1> ;--------------------------------------
7728                            <1>       ; 03/01/2015
7729                            <1>       ; According to ATA-ATAPI specification v2.0 to v5.0
7730                            <1>       ; logical sector per logical track
7731                            <1>       ; and logical heads - 1 would be set but
7732                            <1>       ; it is seen as it will be good
7733                            <1>       ; if physical parameters will be set here
7734                            <1>       ; because, number of heads <= 16.
7735                            <1>       ; (logical heads usually more than 16)
7736                            <1>       ; NOTE: ATA logical parameters (software C, H, S)
7737                            <1>       ;     == INT 13h physical parameters
7738                            <1>
7739                            <1> ;INIT_DRV:
7740                            <1> ;     MOV    byte [CMD_BLOCK+6],SET_PARM_CMD
7741                            <1> ;     CALL   GET_VEC             ; ES:BX -> PARAMETER BLOCK
7742                            <1> ;     MOV    AL,[ES:BX+2]        ; GET NUMBER OF HEADS
7743                            <1> ;     DEC    AL                  ; CONVERT TO 0-INDEX
7744                            <1> ;     MOV    AH,[CMD_BLOCK+5]    ; GET SDH REGISTER
7745                            <1> ;     AND    AH,0F0H             ; CHANGE HEAD NUMBER
7746                            <1> ;     OR     AH,AL               ; TO MAX HEAD
7747                            <1> ;     MOV    [CMD_BLOCK+5],AH
7748                            <1> ;     MOV    AL,[ES:BX+14]       ; MAX SECTOR NUMBER
7749                            <1> ;     MOV    [CMD_BLOCK+1],AL
7750                            <1> ;     SUB    AX,AX
7751                            <1> ;     MOV    [CMD_BLOCK+3],AL    ; ZERO FLAGS
7752                            <1> ;     CALL   COMMAND             ; TELL CONTROLLER
7753                            <1> ;     JNZ    short INIT_EXIT           ; CONTROLLER BUSY ERROR
7754                            <1> ;     CALL   NOT_BUSY            ; WAIT FOR IT TO BE DONE
7755                            <1> ;     JNZ    short INIT_EXIT           ; TIME OUT
7756                            <1> ;     CALL   CHECK_STATUS
7757                            <1> ;INIT_EXIT:
7758                            <1> ;     RETn
7759                            <1>
7760                            <1> ; 04/01/2015
7761                            <1> ; 02/01/2015 - Derived from from AWARD BIOS 1999
7762                            <1> ;                          AHDSK.ASM - INIT_DRIVE
```

```
7763                            <1> INIT_DRV:
7764                            <1>       ;xor   ah,ah
7765 00005648 31C0              <1>       xor    eax, eax ; 21/02/2015
7766 0000564A B00B              <1>       mov    al,11 ; Physical heads from translated HDPT
7767 0000564C 3825[58810100]    <1>       cmp    [LBAMode], ah   ; 0
7768 00005652 7702              <1>       ja     short idrv0
7769 00005654 B002              <1>       mov    al,2  ; Physical heads from standard HDPT
7770                            <1> idrv0:
7771                            <1>       ; DL = drive number (0 based)
7772 00005656 E8C3030000        <1>       call   GET_VEC
7773                            <1>       ;push  bx
7774 0000565B 53                <1>       push   ebx ; 21/02/2015
7775                            <1>       ;add   bx,ax
7776 0000565C 01C3              <1>       add    ebx, eax
7777                            <1>       ;; 05/01/2015
7778 0000565E 8A25[E4680000]    <1>       mov    ah, [hf_m_s] ; drive number (0= master, 1= slave)
7779                            <1>       ;;and  ah,1
7780 00005664 C0E404            <1>       shl    ah,4
7781 00005667 80CCA0            <1>       or     ah,0A0h  ; Drive/Head register - 10100000b (A0h)
7782                            <1>       ;mov   al,[es:bx]
7783 0000566A 8A03              <1>       mov    al, [ebx] ; 21/02/2015
7784 0000566C FEC8              <1>       dec    al       ; last head number
7785                            <1>       ;and   al,0Fh
7786 0000566E 08E0              <1>       or     al,ah   ; lower 4 bits for head number
7787                            <1>       ;
7788 00005670 C645FE91          <1>       mov    byte [CMD_BLOCK+6],SET_PARM_CMD
7789 00005674 8845FD            <1>       mov    [CMD_BLOCK+5],al
7790                            <1>       ;pop   bx
7791 00005677 5B                <1>       pop    ebx
7792 00005678 29C0              <1>       sub    eax, eax ; 21/02/2015
7793 0000567A B004              <1>       mov    al,4 ; Physical sec per track from translated HDPT
7794 0000567C 803D[58810100]00  <1>       cmp    byte [LBAMode], 0
7795 00005683 7702              <1>       ja     short idrv1
7796 00005685 B00E              <1>       mov    al,14 ; Physical sec per track from standard HDPT
7797                            <1> idrv1:
7798                            <1>       ;xor   ah,ah
7799                            <1>       ;add   bx,ax
7800 00005687 01C3              <1>       add    ebx, eax ; 21/02/2015
7801                            <1>       ;mov   al,[es:bx]
7802                            <1>                     ; sector number
7803 00005689 8A03              <1>       mov    al, [ebx]
7804 0000568B 8845F9            <1>       mov    [CMD_BLOCK+1],al
7805 0000568E 28C0              <1>       sub    al,al
7806 00005690 8845FB            <1>       mov    [CMD_BLOCK+3],al  ; ZERO FLAGS
7807 00005693 E8CA010000        <1>       call   COMMAND         ; TELL CONTROLLER
7808 00005698 750C              <1>       jnz    short INIT_EXIT       ; CONTROLLER BUSY ERROR
7809 0000569A E878020000        <1>       call   NOT_BUSY       ; WAIT FOR IT TO BE DONE
7810 0000569F 7505              <1>       jnz    short INIT_EXIT       ; TIME OUT
7811 000056A1 E8C9020000        <1>       call   CHECK_STATUS
7812                            <1> INIT_EXIT:
7813 000056A6 C3                <1>       RETn
7814                            <1>
7815                            <1> ;---------------------------------------
7816                            <1> ;     READ LONG         (AH = 0AH) :
7817                            <1> ;---------------------------------------
7818                            <1>
7819                            <1> RD_LONG:
7820                            <1>       ;MOV   @CMD_BLOCK+6,READ_CMD OR ECC_MODE
7821 000056A7 C645FE22          <1>       mov    byte [CMD_BLOCK+6],READ_CMD + ECC_MODE
7822 000056AB E9E0000000        <1>       JMP   COMMANDI
7823                            <1>
7824                            <1> ;---------------------------------------
7825                            <1> ;     WRITE LONG        (AH = 0BH) :
7826                            <1> ;---------------------------------------
7827                            <1>
7828                            <1> WR_LONG:
7829                            <1>       ;MOV   @CMD_BLOCK+6,WRITE_CMD OR ECC_MODE
7830 000056B0 C645FE32          <1>       MOV   byte [CMD_BLOCK+6],WRITE_CMD + ECC_MODE
7831 000056B4 E932010000        <1>       JMP   COMMANDO
7832                            <1>
7833                            <1> ;---------------------------------------
7834                            <1> ;     SEEK              (AH = 0CH) :
7835                            <1> ;---------------------------------------
7836                            <1>
7837                            <1> DISK_SEEK:
7838 000056B9 C645FE70          <1>       MOV   byte [CMD_BLOCK+6],SEEK_CMD
7839 000056BD E8A0010000        <1>       CALL   COMMAND
7840 000056C2 751C              <1>       JNZ    short DS_EXIT       ; CONTROLLER BUSY ERROR
7841 000056C4 E812020000        <1>       CALL   _WAIT
7842 000056C9 7515              <1>       JNZ   DS_EXIT             ; TIME OUT ON SEEK
7843 000056CB E89F020000        <1>       CALL   CHECK_STATUS
7844 000056D0 803D[43810100]40  <1>       CMP   byte [DISK_STATUS1],BAD_SEEK
7845 000056D7 7507              <1>       JNE    short DS_EXIT
7846 000056D9 C605[43810100]00  <1>       MOV   byte [DISK_STATUS1],0
7847                            <1> DS_EXIT:
7848 000056E0 C3                <1>       RETn
7849                            <1>
7850                            <1> ;---------------------------------------
7851                            <1> ;     TEST DISK READY      (AH = 10H) :
7852                            <1> ;---------------------------------------
7853                            <1>
7854                            <1> TST_RDY:                        ; WAIT FOR CONTROLLER
7855 000056E1 E831020000        <1>       CALL   NOT_BUSY
7856 000056E6 751C              <1>       JNZ    short TR_EX
7857 000056E8 8A45FD            <1>       MOV    AL,[CMD_BLOCK+5]    ; SELECT DRIVE
7858 000056EB 668B15[E0680000]  <1>       MOV    DX,[HF_PORT]
7859 000056F2 80C206            <1>       add    dl,6
7860 000056F5 EE                <1>       OUT    DX,AL
7861 000056F6 E88C020000        <1>       CALL   CHECK_ST            ; CHECK STATUS ONLY
7862 000056FB 7507              <1>       JNZ    short TR_EX
7863 000056FD C605[43810100]00  <1>       MOV    byte [DISK_STATUS1],0 ; WIPE OUT DATA CORRECTED ERROR
7864                            <1> TR_EX:
7865 00005704 C3                <1>       RETn
7866                            <1>
7867                            <1> ;---------------------------------------
```

```
7868                            <1> ;     RECALIBRATE       (AH = 11H) :
7869                            <1> ;----------------------------------------
7870                            <1>
7871                            <1> HDISK_RECAL:
7872 00005705 C645FE10          <1>       MOV    byte [CMD_BLOCK+6],RECAL_CMD ; 10h, 16
7873 00005709 E854010000        <1>       CALL   COMMAND             ; START THE OPERATION
7874 0000570E 7523              <1>       JNZ    short RECAL_EXIT    ; ERROR
7875 00005710 E8C6010000        <1>       CALL   _WAIT               ; WAIT FOR COMPLETION
7876 00005715 7407              <1>       JZ     short RECAL_X       ; TIME OUT ONE OK ?
7877 00005717 E8BF010000        <1>       CALL   _WAIT               ; WAIT FOR COMPLETION LONGER
7878 0000571C 7515              <1>       JNZ    short RECAL_EXIT    ; TIME OUT TWO TIMES IS ERROR
7879                            <1> RECAL_X:
7880 0000571E E84C020000        <1>       CALL   CHECK_STATUS
7881 00005723 803D[43810100]40  <1>       CMP    byte [DISK_STATUS1],BAD_SEEK ; SEEK NOT COMPLETE
7882 0000572A 7507              <1>       JNE    short RECAL_EXIT    ; IS OK
7883 0000572C C605[43810100]00  <1>       MOV    byte [DISK_STATUS1],0
7884                            <1> RECAL_EXIT:
7885 00005733 803D[43810100]00  <1>       CMP     byte [DISK_STATUS1],0
7886 0000573A C3                <1>       RETn
7887                            <1>
7888                            <1> ;----------------------------------------
7889                            <1> ;      CONTROLLER DIAGNOSTIC (AH = 14H) :
7890                            <1> ;----------------------------------------
7891                            <1>
7892                            <1> CTLR_DIAGNOSTIC:
7893 0000573B FA                <1>       CLI                        ; DISABLE INTERRUPTS WHILE CHANGING MASK
7894 0000573C E4A1              <1>       IN     AL,INTB01           ; TURN ON SECOND INTERRUPT CHIP
7895                            <1>       ;AND   AL,0BFH
7896 0000573E 243F              <1>       and    al, 3Fh             ; enable IRQ 14 & IRQ 15
7897                            <1>       ;JMP   $+2
7898                            <1>       IODELAY
2990 00005740 EB00              <2>  jmp short $+2
2991 00005742 EB00              <2>  jmp short $+2
7899 00005744 E6A1              <1>       OUT    INTB01,AL
7900                            <1>       IODELAY
2990 00005746 EB00              <2>  jmp short $+2
2991 00005748 EB00              <2>  jmp short $+2
7901 0000574A E421              <1>       IN     AL,INTA01           ; LET INTERRUPTS PASS THRU TO
7902 0000574C 24FB              <1>       AND    AL,0FBH             ;  SECOND CHIP
7903                            <1>       ;JMP   $+2
7904                            <1>       IODELAY
2990 0000574E EB00              <2>  jmp short $+2
2991 00005750 EB00              <2>  jmp short $+2
7905 00005752 E621              <1>       OUT    INTA01,AL
7906 00005754 FB                <1>       STI
7907 00005755 E8BD010000        <1>       CALL   NOT_BUSY            ; WAIT FOR CARD
7908 0000575A 752B              <1>       JNZ    short CD_ERR        ; BAD CARD
7909                            <1>       ;MOV   DX, HF_PORT+7
7910 0000575C 668B15[E0680000]  <1>       mov    dx, [HF_PORT]
7911 00005763 80C207            <1>       add    dl, 7
7912 00005766 B090              <1>       MOV    AL,DIAG_CMD         ; START DIAGNOSE
7913 00005768 EE                <1>       OUT    DX,AL
7914 00005769 E8A9010000        <1>       CALL   NOT_BUSY            ; WAIT FOR IT TO COMPLETE
7915 0000576E B480              <1>       MOV    AH,TIME_OUT
7916 00005770 7517              <1>       JNZ    short CD_EXIT             ; TIME OUT ON DIAGNOSTIC
7917                            <1>       ;MOV   DX,HF_PORT+1        ; GET ERROR REGISTER
7918 00005772 668B15[E0680000]  <1>       mov    dx, [HF_PORT]
7919 00005779 FEC2              <1>       inc    dl
7920 0000577B EC                <1>       IN     AL,DX
7921 0000577C A2[3A810100]      <1>       MOV    [HF_ERROR],AL       ; SAVE IT
7922 00005781 B400              <1>       MOV    AH,0
7923 00005783 3C01              <1>       CMP    AL,1                ; CHECK FOR ALL OK
7924 00005785 7402              <1>       JE     SHORT CD_EXIT
7925 00005787 B420              <1> CD_ERR: MOV  AH,BAD_CNTLR
7926                            <1> CD_EXIT:
7927 00005789 8825[43810100]    <1>       MOV    [DISK_STATUS1],AH
7928 0000578F C3                <1>       RETn
7929                            <1>
7930                            <1> ;----------------------------------------
7931                            <1> ; COMMANDI                             :
7932                            <1> ;     REPEATEDLY INPUTS DATA TILL      :
7933                            <1> ;     NSECTOR RETURNS ZERO             :
7934                            <1> ;----------------------------------------
7935                            <1> COMMANDI:
7936 00005790 E862020000        <1>       CALL   CHECK_DMA           ; CHECK 64K BOUNDARY ERROR
7937 00005795 7253              <1>       JC     short CMD_ABORT
7938                            <1>       ;MOV   DI,BX
7939 00005797 89DF              <1>       mov    edi, ebx ; 21/02/2015
7940 00005799 E8C4000000        <1>       CALL   COMMAND             ; OUTPUT COMMAND
7941 0000579E 754A              <1>       JNZ    short CMD_ABORT
7942                            <1> CMD_I1:
7943 000057A0 E836010000        <1>       CALL   _WAIT               ; WAIT FOR DATA REQUEST INTERRUPT
7944 000057A5 7543              <1>       JNZ    short TM_OUT        ; TIME OUT
7945                            <1> cmd_i1x: ; 18/02/2016
7946                            <1>       ;MOV   CX,256              ; SECTOR SIZE IN WORDS
7947 000057A7 B900010000        <1>       mov    ecx, 256 ; 21/02/2015
7948                            <1>       ;MOV   DX,HF_PORT
7949 000057AC 668B15[E0680000]  <1>       mov    dx,[HF_PORT]
7950 000057B3 FA                <1>       CLI
7951 000057B4 FC                <1>       CLD
7952 000057B5 F3666D            <1>       REP    INSW                ; GET THE SECTOR
7953 000057B8 FB                <1>       STI
7954 000057B9 F645FE02          <1>       TEST   byte [CMD_BLOCK+6],ECC_MODE ; CHECK FOR NORMAL INPUT
7955 000057BD 7419              <1>       JZ     short CMD_I3
7956 000057BF E880010000        <1>       CALL   WAIT_DRQ            ; WAIT FOR DATA REQUEST
7957 000057C4 7224              <1>       JC     short TM_OUT
7958                            <1>       ;MOV   DX,HF_PORT
7959 000057C6 668B15[E0680000]  <1>       mov    dx,[HF_PORT]
7960                            <1>       ;MOV   CX,4                ; GET ECC BYTES
7961 000057CD B904000000        <1>       mov    ecx, 4 ; mov cx, 4
7962 000057D2 EC                <1> CMD_I2: IN   AL,DX
7963                            <1>       ;MOV   [ES:DI],AL          ; GO SLOW FOR BOARD
7964 000057D3 8807              <1>       mov    [edi], al ; 21/02/2015
7965 000057D5 47                <1>       INC    eDI
7966 000057D6 E2FA              <1>       LOOP   CMD_I2
```

```
                        <1> CMD_I3:
                        <1>       ; wait for 400 ns
000057D8 80C207         <1>       add    dl, 7
000057DB EC             <1>       in     al, dx
000057DC EC             <1>       in     al, dx
000057DD EC             <1>       in     al, dx
                        <1>       ;
000057DE E88C010000     <1>       CALL   CHECK_STATUS
000057E3 7505           <1>       JNZ    short CMD_ABORT     ; ERROR RETURNED
000057E5 FE4DF9         <1>       DEC    byte [CMD_BLOCK+1]  ; CHECK FOR MORE
                        <1>       ;JNZ   SHORT CMD_I1
000057E8 75BD           <1>       jnz    short cmd_i1x ; 18/02/2016
                        <1> CMD_ABORT:
000057EA C3             <1> TM_OUT: RETn
                        <1>
                        <1> ;---------------------------------------
                        <1> ; COMMANDO                            :
                        <1> ;     REPEATEDLY OUTPUTS DATA TILL    :
                        <1> ;     NSECTOR RETURNS ZERO            :
                        <1> ;---------------------------------------
                        <1> COMMANDO:
000057EB E807020000     <1>       CALL   CHECK_DMA          ; CHECK 64K BOUNDARY ERROR
000057F0 72F8           <1>       JC     short CMD_ABORT
000057F2 89DE           <1> CMD_OF: MOV  eSI,eBX ; 21/02/2015
000057F4 E869000000     <1>       CALL   COMMAND            ; OUTPUT COMMAND
000057F9 75EF           <1>       JNZ    short CMD_ABORT
000057FB E844010000     <1>       CALL   WAIT_DRQ           ; WAIT FOR DATA REQUEST
00005800 72E8           <1>       JC     short TM_OUT             ; TOO LONG
                        <1> CMD_O1: ;PUSH      DS
                        <1>       ;PUSH  ES                 ; MOVE ES TO DS
                        <1>       ;POP   DS
                        <1>       ;MOV   CX,256             ; PUT THE DATA OUT TO THE CARD
                        <1>       ;MOV   DX,HF_PORT
                        <1>       ; 01/02/2015
00005802 668B15[E0680000]   <1>       mov    dx, [HF_PORT]
                        <1>       ;push  es
                        <1>       ;pop   ds
                        <1>       ;mov   cx, 256
00005809 B900010000     <1>       mov    ecx, 256 ; 21/02/2015
0000580E FA             <1>       CLI
0000580F FC             <1>       CLD
00005810 F3666F         <1>       REP    OUTSW
00005813 FB             <1>       STI
                        <1>       ;POP   DS                 ; RESTORE DS
00005814 F645FE02       <1>       TEST   byte [CMD_BLOCK+6],ECC_MODE ; CHECK FOR NORMAL OUTPUT
00005818 7419           <1>       JZ     short CMD_O3
0000581A E825010000     <1>       CALL   WAIT_DRQ           ; WAIT FOR DATA REQUEST
0000581F 72C9           <1>       JC     short TM_OUT
                        <1>       ;MOV   DX,HF_PORT
00005821 668B15[E0680000]   <1>       mov    dx, [HF_PORT]
                        <1>       ;MOV   CX,4               ; OUTPUT THE ECC BYTES
00005828 B904000000     <1>       mov    ecx, 4  ; mov cx, 4
                        <1> CMD_O2: ;MOV AL,[ES:SI]
0000582D 8A06           <1>       mov    al, [esi]
0000582F EE             <1>       OUT    DX,AL
00005830 46             <1>       INC    eSI
00005831 E2FA           <1>       LOOP   CMD_O2
                        <1> CMD_O3:
00005833 E8A3000000     <1>       CALL   _WAIT              ; WAIT FOR SECTOR COMPLETE INTERRUPT
00005838 75B0           <1>       JNZ    short TM_OUT       ; ERROR RETURNED
0000583A E830010000     <1>       CALL   CHECK_STATUS
0000583F 75A9           <1>       JNZ    short CMD_ABORT
00005841 F605[39810100]08   <1>       TEST   byte [HF_STATUS],ST_DRQ ; CHECK FOR MORE
00005848 75B8           <1>       JNZ    SHORT CMD_O1
                        <1>       ;MOV   DX,HF_PORT+2       ; CHECK RESIDUAL SECTOR COUNT
0000584A 668B15[E0680000]   <1>       mov    dx, [HF_PORT]
                        <1>       ;add   dl, 2
00005851 FEC2           <1>       inc    dl
00005853 FEC2           <1>       inc    dl
00005855 EC             <1>       IN     AL,DX              ;
00005856 A8FF           <1>       TEST   AL,0FFH            ;
00005858 7407           <1>       JZ     short CMD_O4       ; COUNT = 0  OK
0000585A C605[43810100]BB   <1>       MOV    byte [DISK_STATUS1],UNDEF_ERR
                        <1>                                 ; OPERATION ABORTED - PARTIAL TRANSFER
                        <1> CMD_O4:
00005861 C3             <1>       RETn
                        <1>
                        <1> ;---------------------------------------------------------
                        <1> ; COMMAND                                               :
                        <1> ;     THIS ROUTINE OUTPUTS THE COMMAND BLOCK            :
                        <1> ; OUTPUT                                                :
                        <1> ;     BL = STATUS                                       :
                        <1> ;     BH = ERROR REGISTER                               :
                        <1> ;---------------------------------------------------------
                        <1>
                        <1> COMMAND:
00005862 53             <1>       PUSH   eBX                ; WAIT FOR SEEK COMPLETE AND READY
                        <1>       ;;MOV  CX,DELAY_2         ; SET INITIAL DELAY BEFORE TEST
                        <1> COMMAND1:
                        <1>       ;;PUSH CX                 ; SAVE LOOP COUNT
00005863 E879FEFFFF     <1>       CALL   TST_RDY            ; CHECK DRIVE READY
                        <1>       ;;POP  CX
00005868 7419           <1>       JZ     short COMMAND2     ; DRIVE IS READY
0000586A 803D[43810100]80   <1>       CMP   byte [DISK_STATUS1],TIME_OUT ; TST_RDY TIMED OUT--GIVE UP
                        <1>       ;JZ    short CMD_TIMEOUT
                        <1>       ;;LOOP COMMAND1           ; KEEP TRYING FOR A WHILE
                        <1>       ;JMP   SHORT COMMAND4     ; ITS NOT GOING TO GET READY
00005871 7507           <1>       jne    short COMMAND4
                        <1> CMD_TIMEOUT:
00005873 C605[43810100]20   <1>       MOV    byte [DISK_STATUS1],BAD_CNTLR
                        <1> COMMAND4:
0000587A 5B             <1>       POP    eBX
0000587B 803D[43810100]00   <1>       CMP   byte [DISK_STATUS1],0  ; SET CONDITION CODE FOR CALLER
00005882 C3             <1>       RETn
                        <1> COMMAND2:
```

```
8072 00005883 5B                <1>       POP    eBX
8073 00005884 57                <1>       PUSH   eDI
8074 00005885 C605[3B810100]00  <1>       MOV    byte [HF_INT_FLAG],0 ; RESET INTERRUPT FLAG
8075 0000588C FA                <1>       CLI                         ; INHIBIT INTERRUPTS WHILE CHANGING MASK
8076 0000588D E4A1              <1>       IN     AL,INTB01            ; TURN ON SECOND INTERRUPT CHIP
8077                            <1>       ;AND   AL,0BFH
8078 0000588F 243F              <1>       and    al, 3Fh              ; Enable IRQ 14 & 15
8079                            <1>       ;JMP   $+2
8080                            <1>       IODELAY
2990 00005891 EB00              <2>  jmp short $+2
2991 00005893 EB00              <2>  jmp short $+2
8081 00005895 E6A1              <1>       OUT    INTB01,AL
8082 00005897 E421              <1>       IN     AL,INTA01            ; LET INTERRUPTS PASS THRU TO
8083 00005899 24FB              <1>       AND    AL,0FBH              ;  SECOND CHIP
8084                            <1>       ;JMP   $+2
8085                            <1>       IODELAY
2990 0000589B EB00              <2>  jmp short $+2
2991 0000589D EB00              <2>  jmp short $+2
8086 0000589F E621              <1>       OUT    INTA01,AL
8087 000058A1 FB                <1>       STI
8088 000058A2 31FF              <1>       XOR    eDI,eDI              ; INDEX THE COMMAND TABLE
8089                            <1>       ;MOV   DX,HF_PORT+1         ; DISK ADDRESS
8090 000058A4 668B15[E0680000]  <1>       mov    dx, [HF_PORT]
8091 000058AB FEC2              <1>       inc    dl
8092 000058AD F605[45810100]C0  <1>       TEST   byte [CONTROL_BYTE],0C0H ; CHECK FOR RETRY SUPPRESSION
8093 000058B4 7411              <1>       JZ     short COMMAND3
8094 000058B6 8A45FE            <1>       MOV    AL, [CMD_BLOCK+6]    ; YES-GET OPERATION CODE
8095 000058B9 24F0              <1>       AND    AL,0F0H              ; GET RID OF MODIFIERS
8096 000058BB 3C20              <1>       CMP    AL,20H               ; 20H-40H IS READ, WRITE, VERIFY
8097 000058BD 7208              <1>       JB     short COMMAND3
8098 000058BF 3C40              <1>       CMP    AL,40H
8099 000058C1 7704              <1>       JA     short COMMAND3
8100 000058C3 804DFE01          <1>       OR     byte [CMD_BLOCK+6],NO_RETRIES
8101                            <1>                                    ; VALID OPERATION FOR RETRY SUPPRESS
8102                            <1> COMMAND3:
8103 000058C7 8A443DF8          <1>       MOV    AL,[CMD_BLOCK+eDI]   ; GET THE COMMAND STRING BYTE
8104 000058CB EE                <1>       OUT    DX,AL                ; GIVE IT TO CONTROLLER
8105                            <1>       IODELAY
2990 000058CC EB00              <2>  jmp short $+2
2991 000058CE EB00              <2>  jmp short $+2
8106 000058D0 47                <1>       INC    eDI                  ; NEXT BYTE IN COMMAND BLOCK
8107 000058D1 6642              <1>       INC    DX                   ; NEXT DISK ADAPTER REGISTER
8108 000058D3 6683FF07          <1>       cmp    di, 7  ; 1/1/2015    ; ALL DONE?
8109 000058D7 75EE              <1>       JNZ    short COMMAND3       ; NO--GO DO NEXT ONE
8110 000058D9 5F                <1>       POP    eDI
8111 000058DA C3                <1>       RETn                        ; ZERO FLAG IS SET
8112                            <1>
8113                            <1> ;CMD_TIMEOUT:
8114                            <1> ;     MOV    byte [DISK_STATUS1],BAD_CNTLR
8115                            <1> ;COMMAND4:
8116                            <1> ;     POP    BX
8117                            <1> ;     CMP    [DISK_STATUS1],0     ; SET CONDITION CODE FOR CALLER
8118                            <1> ;     RETn
8119                            <1>
8120                            <1> ;----------------------------------------
8121                            <1> ;     WAIT FOR INTERRUPT        :
8122                            <1> ;----------------------------------------
8123                            <1> ;WAIT:
8124                            <1> _WAIT:
8125 000058DB FB                <1>       STI                         ; MAKE SURE INTERRUPTS ARE ON
8126                            <1>       ;SUB   CX,CX                ; SET INITIAL DELAY BEFORE TEST
8127                            <1>       ;CLC
8128                            <1>       ;MOV   AX,9000H             ; DEVICE WAIT INTERRUPT
8129                            <1>       ;INT   15H
8130                            <1>       ;JC    WT2                  ; DEVICE TIMED OUT
8131                            <1>       ;MOV   BL,DELAY_1           ; SET DELAY COUNT
8132                            <1>
8133                            <1>       ;mov   bl, WAIT_HDU_INT_HI
8134                            <1>       ;; 21/02/2015
8135                            <1>       ;;mov  bl, WAIT_HDU_INT_HI + 1
8136                            <1>       ;;mov  cx, WAIT_HDU_INT_LO
8137 000058DC B915160500          <1>       mov    ecx, WAIT_HDU_INT_LH
8138                            <1>                                    ; (AWARD BIOS -> WAIT_FOR_MEM)
8139                            <1> ;----- WAIT LOOP
8140                            <1>
8141                            <1> WT1:
8142                            <1>       ;TEST  byte [HF_INT_FLAG],80H ; TEST FOR INTERRUPT
8143 000058E1 F605[3B810100]C0  <1>       test   byte [HF_INT_FLAG],0C0h
8144                            <1>       ;LOOPZ WT1
8145 000058E8 7517              <1>       JNZ    short WT3            ; INTERRUPT--LETS GO
8146                            <1>       ;DEC   BL
8147                            <1>       ;JNZ   short WT1            ; KEEP TRYING FOR A WHILE
8148                            <1>
8149                            <1> WT1_hi:
8150 000058EA E461              <1>       in     al, SYS1 ; 61h (PORT_B) ; wait for lo to hi
8151 000058EC A810              <1>       test   al, 10h                 ; transition on memory
8152 000058EE 75FA              <1>       jnz    short WT1_hi         ; refresh.
8153                            <1> WT1_lo:
8154 000058F0 E461              <1>       in     al, SYS1             ; 061h (PORT_B)
8155 000058F2 A810              <1>       test   al, 10h
8156 000058F4 74FA              <1>       jz     short WT1_lo
8157 000058F6 E2E9              <1>       loop   WT1
8158                            <1>       ;;or   bl, bl
8159                            <1>       ;;jz   short WT2
8160                            <1>       ;;dec  bl
8161                            <1>       ;;jmp  short WT1
8162                            <1>       ;dec   bl
8163                            <1>       ;jnz   short WT1
8164                            <1>
8165 000058F8 C605[43810100]80  <1> WT2:  MOV    byte [DISK_STATUS1],TIME_OUT ; REPORT TIME OUT ERROR
8166 000058FF EB0E              <1>       JMP    SHORT WT4
8167 00005901 C605[43810100]00  <1> WT3:  MOV    byte [DISK_STATUS1],0
8168 00005908 C605[3B810100]00  <1>       MOV    byte [HF_INT_FLAG],0
8169 0000590F 803D[43810100]00  <1> WT4:  CMP    byte [DISK_STATUS1],0     ; SET CONDITION CODE FOR CALLER
8170 00005916 C3                <1>       RETn
```

```
                            <1>
                            <1> ;----------------------------------------
                            <1> ;     WAIT FOR CONTROLLER NOT BUSY     :
                            <1> ;----------------------------------------
                            <1> NOT_BUSY:
00005917 FB                 <1>       STI                         ; MAKE SURE INTERRUPTS ARE ON
                            <1>       ;PUSH  eBX
                            <1>       ;SUB   CX,CX                ; SET INITIAL DELAY BEFORE TEST
00005918 668B15[E0680000]   <1>       mov    DX, [HF_PORT]
0000591F 80C207             <1>       add    dl, 7                ; Status port (HF_PORT+7)
                            <1>       ;MOV   BL,DELAY_1
                            <1>                                   ; wait for 10 seconds
                            <1>       ;mov   cx, WAIT_HDU_INT_LO ; 1615h
                            <1>       ;;mov  bl, WAIT_HDU_INT_HI ;   05h
                            <1>       ;mov   bl, WAIT_HDU_INT_HI + 1
00005922 B915160500         <1>       mov    ecx, WAIT_HDU_INT_LH  ; 21/02/2015
                            <1>       ;
                            <1> ;;    mov     byte [wait_count], 0 ; Reset wait counter
                            <1> NB1:
00005927 EC                 <1>       IN     AL,DX                ; CHECK STATUS
                            <1>       ;TEST  AL,ST_BUSY
00005928 2480               <1>       and    al, ST_BUSY
                            <1>       ;LOOPNZ NB1
0000592A 7410               <1>       JZ     short NB2            ; NOT BUSY--LETS GO
                            <1>       ;DEC   BL
                            <1>       ;JNZ   short NB1            ; KEEP TRYING FOR A WHILE
                            <1>
0000592C E461               <1> NB1_hi: IN   AL,SYS1               ; wait for hi to lo
0000592E A810               <1>       TEST   AL,010H              ; transition on memory
00005930 75FA               <1>       JNZ    SHORT NB1_hi         ; refresh.
00005932 E461               <1> NB1_lo: IN   AL,SYS1
00005934 A810               <1>       TEST   AL,010H
00005936 74FA               <1>       JZ     short NB1_lo
00005938 E2ED               <1>       LOOP   NB1
                            <1>       ;dec   bl
                            <1>       ;jnz   short NB1
                            <1>       ;
                            <1> ;;    cmp   byte [wait_count], 182  ; 10 seconds (182 timer ticks)
                            <1> ;;    jb     short NB1
                            <1>       ;
                            <1>       ;MOV   [DISK_STATUS1],TIME_OUT   ; REPORT TIME OUT ERROR
                            <1>       ;JMP   SHORT NB3
0000593A B080               <1>       mov    al, TIME_OUT
                            <1> NB2:
                            <1>       ;MOV   byte [DISK_STATUS1],0
                            <1> ;NB3:
                            <1>       ;POP   eBX
0000593C A2[43810100]       <1>       mov    [DISK_STATUS1], al  ;;; will be set after return
                            <1>       ;CMP   byte [DISK_STATUS1],0 ; SET CONDITION CODE FOR CALLER
00005941 08C0               <1>       or     al, al               ; (zf = 0 --> timeout)
00005943 C3                 <1>       RETn
                            <1>
                            <1> ;----------------------------------------
                            <1> ;     WAIT FOR DATA REQUEST            :
                            <1> ;----------------------------------------
                            <1> WAIT_DRQ:
                            <1>       ;MOV   CX,DELAY_3
                            <1>       ;MOV   DX,HF_PORT+7
00005944 668B15[E0680000]   <1>       mov    dx, [HF_PORT]
0000594B 80C207             <1>       add    dl, 7
                            <1>       ;;MOV  bl, WAIT_HDU_DRQ_HI ; 0
                            <1>       ;MOV   cx, WAIT_HDU_DRQ_LO ; 1000 (30 milli seconds)
                            <1>                                   ; (but it is written as 2000
                            <1>                                   ; micro seconds in ATORGS.ASM file
                            <1>                                   ; of Award Bios - 1999, D1A0622)
0000594E B9E8030000         <1>       mov    ecx, WAIT_HDU_DRQ_LH ; 21/02/2015
00005953 EC                 <1> WQ_1: IN     AL,DX                ; GET STATUS
00005954 A808               <1>       TEST   AL,ST_DRQ            ; WAIT FOR DRQ
00005956 7516               <1>       JNZ    short WQ_OK
                            <1>       ;LOOP  WQ_1                 ; KEEP TRYING FOR A SHORT WHILE
                            <1> WQ_hi:
00005958 E461               <1>       IN     AL,SYS1              ; wait for hi to lo
0000595A A810               <1>       TEST   AL,010H              ; transition on memory
0000595C 75FA               <1>       JNZ    SHORT WQ_hi          ; refresh.
0000595E E461               <1> WQ_lo:  IN      AL,SYS1
00005960 A810               <1>       TEST   AL,010H
00005962 74FA               <1>       JZ     SHORT WQ_lo
00005964 E2ED               <1>       LOOP   WQ_1
                            <1>
00005966 C605[43810100]80   <1>       MOV    byte [DISK_STATUS1],TIME_OUT ; ERROR
0000596D F9                 <1>       STC
                            <1> WQ_OK:
0000596E C3                 <1>       RETn
                            <1> ;WQ_OK:      ;CLC
                            <1> ;     RETn
                            <1>
                            <1> ;----------------------------------------
                            <1> ;     CHECK FIXED DISK STATUS    :
                            <1> ;----------------------------------------
                            <1> CHECK_STATUS:
0000596F E813000000         <1>       CALL   CHECK_ST             ; CHECK THE STATUS BYTE
00005974 7509               <1>       JNZ    short CHECK_S1       ; AN ERROR WAS FOUND
00005976 A801               <1>       TEST   AL,ST_ERROR          ; WERE THERE ANY OTHER ERRORS
00005978 7405               <1>       JZ     short CHECK_S1       ; NO ERROR REPORTED
0000597A E849000000         <1>       CALL   CHECK_ER             ; ERROR REPORTED
                            <1> CHECK_S1:
0000597F 803D[43810100]00   <1>       CMP    byte [DISK_STATUS1],0 ; SET STATUS FOR CALLER
00005986 C3                 <1>       RETn
                            <1>
                            <1> ;----------------------------------------
                            <1> ;     CHECK FIXED DISK STATUS BYTE     :
                            <1> ;----------------------------------------
                            <1> CHECK_ST:
                            <1>       ;MOV   DX,HF_PORT+7         ; GET THE STATUS
00005987 668B15[E0680000]   <1>       mov    dx, [HF_PORT]
```

```
8276 0000598E 80C207              <1>       add   dl, 7
8277                              <1>
8278                              <1>       ; 17/02/2016
8279                              <1>       ;(http://wiki.osdev.org/ATA_PIO_Mode)
8280                              <1>       ;"delay 400ns to allow drive to set new values of BSY and DRQ"
8281 00005991 EC                  <1>       IN    AL,DX
8282                              <1>       ;in   al, dx ; 100ns
8283                              <1>       ;in   al, dx ; 100ns
8284                              <1>       ;in   al, dx ; 100ns
8285                              <1>       NEWIODELAY ; 18/02/2016 (AWARD BIOS - 1999, 'CKST' in AHSDK.ASM)
2995 00005992 E6EB                <2>  out 0ebh,al
8286                              <1>       ;
8287 00005994 A2[39810100]        <1>       MOV   [HF_STATUS],AL
8288 00005999 B400                <1>       MOV   AH,0
8289 0000599B A880                <1>       TEST  AL,ST_BUSY          ; IF STILL BUSY
8290 0000599D 751A                <1>       JNZ   short CKST_EXIT           ;  REPORT OK
8291 0000599F B4CC                <1>       MOV   AH,WRITE_FAULT
8292 000059A1 A820                <1>       TEST  AL,ST_WRT_FLT       ; CHECK FOR WRITE FAULT
8293 000059A3 7514                <1>       JNZ   short CKST_EXIT
8294 000059A5 B4AA                <1>       MOV   AH,NOT_RDY
8295 000059A7 A840                <1>       TEST  AL,ST_READY         ; CHECK FOR NOT READY
8296 000059A9 740E                <1>       JZ    short CKST_EXIT
8297 000059AB B440                <1>       MOV   AH,BAD_SEEK
8298 000059AD A810                <1>       TEST  AL,ST_SEEK_COMPL    ; CHECK FOR SEEK NOT COMPLETE
8299 000059AF 7408                <1>       JZ    short CKST_EXIT
8300 000059B1 B411                <1>       MOV   AH,DATA_CORRECTED
8301 000059B3 A804                <1>       TEST  AL,ST_CORRCTD       ; CHECK FOR CORRECTED ECC
8302 000059B5 7502                <1>       JNZ   short CKST_EXIT
8303 000059B7 B400                <1>       MOV   AH,0
8304                              <1> CKST_EXIT:
8305 000059B9 8825[43810100]      <1>       MOV   [DISK_STATUS1],AH   ; SET ERROR FLAG
8306 000059BF 80FC11              <1>       CMP   AH,DATA_CORRECTED   ; KEEP GOING WITH DATA CORRECTED
8307 000059C2 7403                <1>       JZ    short CKST_EX1
8308 000059C4 80FC00              <1>       CMP   AH,0
8309                              <1> CKST_EX1:
8310 000059C7 C3                  <1>       RETn
8311                              <1>
8312                              <1> ;---------------------------------------
8313                              <1> ;     CHECK FIXED DISK ERROR REGISTER :
8314                              <1> ;---------------------------------------
8315                              <1> CHECK_ER:
8316                              <1>       ;MOV   DX, HF_PORT+1      ; GET THE ERROR REGISTER
8317 000059C8 668B15[E0680000]    <1>       mov   dx, [HF_PORT]       ;
8318 000059CF FEC2                <1>       inc   dl
8319 000059D1 EC                  <1>       IN    AL,DX
8320 000059D2 A2[3A810100]        <1>       MOV   [HF_ERROR],AL
8321 000059D7 53                  <1>       PUSH  eBX ; 21/02/2015
8322 000059D8 B908000000          <1>       MOV   eCX,8               ; TEST ALL 8 BITS
8323 000059DD D0E0                <1> CK1:  SHL   AL,1                ; MOVE NEXT ERROR BIT TO CARRY
8324 000059DF 7202                <1>       JC    short CK2           ; FOUND THE ERROR
8325 000059E1 E2FA                <1>       LOOP  CK1                 ; KEEP TRYING
8326 000059E3 BB[D4680000]        <1> CK2:  MOV   eBX, ERR_TBL        ; COMPUTE ADDRESS OF
8327 000059E8 01CB                <1>       ADD   eBX,eCX             ; ERROR CODE
8328                              <1>       ;;MOV  AH,BYTE [CS:BX]    ; GET ERROR CODE
8329                              <1>       ;mov   ah, [bx]
8330 000059EA 8A23                <1>       mov   ah, [ebx] ; 21/02/2015
8331 000059EC 8825[43810100]      <1> CKEX: MOV   [DISK_STATUS1],AH   ; SAVE ERROR CODE
8332 000059F2 5B                  <1>       POP   eBX
8333 000059F3 80FC00              <1>       CMP   AH,0
8334 000059F6 C3                  <1>       RETn
8335                              <1>
8336                              <1> ;--------------------------------------------------------
8337                              <1> ; CHECK_DMA                                           :
8338                              <1> ;  -CHECK ES:BX AND # SECTORS TO MAKE SURE THAT IT WILL   :
8339                              <1> ;   FIT WITHOUT SEGMENT OVERFLOW.                        :
8340                              <1> ;  -ES:BX HAS BEEN REVISED TO THE FORMAT SSSS:000X       :
8341                              <1> ;  -OK IF # SECTORS < 80H (7FH IF LONG READ OR WRITE)    :
8342                              <1> ;  -OK IF # SECTORS = 80H (7FH) AND BX <= 00H (04H)      :
8343                              <1> ;  -ERROR OTHERWISE                                      :
8344                              <1> ;--------------------------------------------------------
8345                              <1> CHECK_DMA:
8346                              <1>       ; 11/04/2021 (32 bit push/pop)
8347 000059F7 50                  <1>       PUSH  eAX                 ; SAVE REGISTERS
8348 000059F8 66B80080            <1>       MOV   AX,8000H            ; AH = MAX # SECTORS AL = MAX OFFSET
8349 000059FC F645FE02            <1>       TEST  byte [CMD_BLOCK+6],ECC_MODE
8350 00005A00 7404                <1>       JZ    short CKD1
8351 00005A02 66B8047F            <1>       MOV   AX,7F04H            ; ECC IS 4 MORE BYTES
8352 00005A06 3A65F9              <1> CKD1: CMP   AH, [CMD_BLOCK+1]   ; NUMBER OF SECTORS
8353 00005A09 7706                <1>       JA    short CKDOK         ; IT WILL FIT
8354 00005A0B 7207                <1>       JB    short CKDERR        ; TOO MANY
8355 00005A0D 38D8                <1>       CMP   AL,BL               ; CHECK OFFSET ON MAX SECTORS
8356 00005A0F 7203                <1>       JB    short CKDERR        ; ERROR
8357 00005A11 F8                  <1> CKDOK: CLC                      ; CLEAR CARRY
8358 00005A12 58                  <1>       POP   eAX
8359 00005A13 C3                  <1>       RETn                      ; NORMAL RETURN
8360 00005A14 F9                  <1> CKDERR: STC                     ; INDICATE ERROR
8361 00005A15 C605[43810100]09    <1>       MOV   byte [DISK_STATUS1],DMA_BOUNDARY
8362 00005A1C 58                  <1>       POP   eAX
8363 00005A1D C3                  <1>       RETn
8364                              <1>
8365                              <1> ;---------------------------------------
8366                              <1> ;     SET UP ES:BX-> DISK PARMS :
8367                              <1> ;---------------------------------------
8368                              <1>
8369                              <1> ; INPUT -> DL = 0 based drive number
8370                              <1> ; OUTPUT -> ES:BX = disk parameter table address
8371                              <1>
8372                              <1> GET_VEC:
8373                              <1>       ;SUB   AX,AX              ; GET DISK PARAMETER ADDRESS
8374                              <1>       ;MOV   ES,AX
8375                              <1>       ;TEST  DL,1
8376                              <1>       ;JZ    short GV_0
8377                              <1> ;     LES    BX,[HF1_TBL_VEC]   ; ES:BX -> DRIVE PARAMETERS
8378                              <1> ;     JMP    SHORT GV_EXIT
8379                              <1> ;GV_0:
```

```
8380                            <1> ;       LES     BX,[HF_TBL_VEC]     ; ES:BX -> DRIVE PARAMETERS
8381                            <1> ;
8382                            <1>         ;xor    bh, bh
8383 00005A1E 31DB             <1>         xor     ebx, ebx
8384 00005A20 88D3             <1>         mov     bl, dl
8385                            <1>         ;;02/01/2015
8386                            <1>         ;;shl   bl, 1               ; port address offset
8387                            <1>         ;;mov   ax, [bx+hd_ports]   ; Base port address (1F0h, 170h)
8388                            <1>         ;;shl   bl, 1               ; dpt pointer offset
8389 00005A22 C0E302           <1>         shl     bl, 2  ;;
8390                            <1>         ;add    bx, HF_TBL_VEC      ; Disk parameter table pointer
8391 00005A25 81C3[48810100]    <1>         add     ebx, HF_TBL_VEC ; 21/02/2015
8392                            <1>         ;push   word [bx+2]         ; dpt segment
8393                            <1>         ;pop    es
8394                            <1>         ;mov    bx, [bx]            ; dpt offset
8395 00005A2B 8B1B             <1>         mov     ebx, [ebx]
8396                            <1> ;GV_EXIT:
8397 00005A2D C3               <1>         RETn
8398                            <1>
8399                            <1> hdc1_int: ; 21/02/2015
8400                            <1> ;--- HARDWARE INT 76H -- ( IRQ LEVEL  14 ) ------------------------
8401                            <1> ;                                                                  :
8402                            <1> ;     FIXED DISK INTERRUPT ROUTINE                                 :
8403                            <1> ;                                                                  :
8404                            <1> ;------------------------------------------------------------------
8405                            <1>
8406                            <1> ; 22/12/2014
8407                            <1> ; IBM PC-XT Model 286 System BIOS Source Code - DISK.ASM (HD_INT)
8408                            <1> ;       '11/15/85'
8409                            <1> ; AWARD BIOS 1999 (D1A0622)
8410                            <1> ;     Source Code - ATORGS.ASM (INT_HDISK, INT_HDISK1)
8411                            <1>
8412                            <1> ;int 76h:
8413                            <1> HD_INT:
8414                            <1>         ; 11/04/2021 (32 bit push/pop)
8415 00005A2E 50               <1>         PUSH   eAX
8416 00005A2F 1E               <1>         PUSH   DS
8417                            <1>         ;CALL  DDS
8418                            <1>         ; 21/02/2015 (32 bit, 386 pm modification)
8419 00005A30 66B81000         <1>         mov    ax, KDATA
8420 00005A34 8ED8             <1>         mov    ds, ax
8421                            <1>         ;
8422                            <1>         ;;MOV  @HF_INT_FLAG,0FFH   ; ALL DONE
8423                            <1>           ;mov byte [CS:HF_INT_FLAG], 0FFh
8424 00005A36 C605[3B810100]FF  <1>         mov    byte [HF_INT_FLAG], 0FFh
8425                            <1>         ;
8426 00005A3D 52               <1>         push   Edx
8427 00005A3E 66BAF701         <1>         mov    dx, HDC1_BASEPORT+7 ; Status Register (1F7h)
8428                            <1>                                    ; Clear Controller
8429                            <1> Clear_IRQ1415:                     ; (Award BIOS - 1999)
8430 00005A42 EC               <1>         in     al, dx              ;
8431 00005A43 5A               <1>         pop    Edx
8432                            <1>         NEWIODELAY
2995 00005A44 E6EB              <2>  out 0ebh,al
8433                            <1>         ;
8434 00005A46 B020             <1>         MOV    AL,EOI              ; NON-SPECIFIC END OF INTERRUPT
8435 00005A48 E6A0             <1>         OUT    INTB00,AL           ; FOR CONTROLLER #2
8436                            <1>         ;JMP   $+2                 ; WAIT
8437                            <1>         NEWIODELAY
2995 00005A4A E6EB              <2>  out 0ebh,al
8438 00005A4C E620             <1>         OUT    INTA00,AL           ; FOR CONTROLLER #1
8439 00005A4E 1F               <1>         POP    DS
8440                            <1>         ;STI                       ; RE-ENABLE INTERRUPTS
8441                            <1>         ;MOV   AX,9100H            ; DEVICE POST
8442                            <1>         ;INT   15H                 ;  INTERRUPT
8443                            <1> irq15_iret: ; 25/02/2015
8444 00005A4F 58               <1>         POP    eAX
8445 00005A50 CF               <1>         IRETd                      ; RETURN FROM INTERRUPT
8446                            <1>
8447                            <1> hdc2_int: ; 21/02/2015
8448                            <1> ;++++ HARDWARE INT 77H ++ ( IRQ LEVEL  15 ) ++++++++++++++++++++
8449                            <1> ;                                                                  :
8450                            <1> ;     FIXED DISK INTERRUPT ROUTINE                                 :
8451                            <1> ;                                                                  :
8452                            <1> ;++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
8453                            <1>
8454                            <1> ;int 77h:
8455                            <1> HD1_INT:
8456                            <1>         ; 11/04/2021 (32 bit push/pop)
8457 00005A51 50               <1>         PUSH   eAX
8458                            <1>         ; Check if that is a spurious IRQ (from slave PIC)
8459                            <1>         ; 25/02/2015 (source: http://wiki.osdev.org/8259_PIC)
8460 00005A52 B00B             <1>         mov    al, 0Bh  ; In-Service Register
8461 00005A54 E6A0             <1>         out    0A0h, al
8462 00005A56 EB00             <1>           jmp short $+2
8463 00005A58 EB00             <1>         jmp short $+2
8464 00005A5A E4A0             <1>         in     al, 0A0h
8465 00005A5C 2480             <1>         and    al, 80h ; bit 7 (is it real IRQ 15 or fake?)
8466 00005A5E 74EF             <1>         jz     short irq15_iret ; Fake (spurious) IRQ, do not send EOI)
8467                            <1>         ;
8468 00005A60 1E               <1>         PUSH   DS
8469                            <1>         ;CALL  DDS
8470                            <1>         ; 21/02/2015 (32 bit, 386 pm modification)
8471 00005A61 66B81000         <1>         mov    ax, KDATA
8472 00005A65 8ED8             <1>         mov    ds, ax
8473                            <1>         ;
8474                            <1>         ;;MOV  @HF_INT_FLAG,0FFH   ; ALL DONE
8475                            <1>           ;or  byte [CS:HF_INT_FLAG],0C0h
8476 00005A67 800D[3B810100]C0  <1>         or     byte [HF_INT_FLAG], 0C0h
8477                            <1>         ;
8478 00005A6E 52               <1>         push   Edx
8479 00005A6F 66BA7701         <1>         mov    dx, HDC2_BASEPORT+7 ; Status Register (177h)
8480                            <1>                                    ; Clear Controller (Award BIOS 1999)
8481 00005A73 EBCD             <1>         jmp    short Clear_IRQ1415
8482                            <1>
```

```
8483                            <1> ;%include 'diskdata.inc' ; 11/03/2015
8484                            <1> ;%include 'diskbss.inc' ; 11/03/2015
8485                            <1>
8486                            <1> ;////////////////////////////////////////////////////////////////
8487                            <1> ;; END OF DISK I/O SYTEM ///
2914                                %include 'memory.s'  ; 09/03/2015
2915                            <1> ; ****************************************************************************
2916                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.4 - memory.s
2917                            <1> ; ----------------------------------------------------------------------------
2918                            <1> ; Last Update: 17/04/2021
2919                            <1> ; ----------------------------------------------------------------------------
2920                            <1> ; Beginning: 24/01/2016
2921                            <1> ; ----------------------------------------------------------------------------
2922                            <1> ; Assembler: NASM version 2.15 (trdos386.s)
2923                            <1> ; ----------------------------------------------------------------------------
2924                            <1> ; Turkish Rational DOS
2925                            <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
2926                            <1> ;
2927                            <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
2928                            <1> ; memory.inc (18/10/2015)
2929                            <1> ; ****************************************************************************
2930                            <1>
2931                            <1> ; MEMORY.ASM - Retro UNIX 386 v1 MEMORY MANAGEMENT FUNCTIONS (PROCEDURES)
2932                            <1> ; Retro UNIX 386 v1 Kernel (unix386.s, v0.2.0.14) - MEMORY.INC
2933                            <1> ; Last Modification: 18/10/2015
2934                            <1>
2935                            <1> ; ///////// MEMORY MANAGEMENT FUNCTIONS (PROCEDURES) ///////////////
2936                            <1>
2937                            <1> ;;04/11/2014 (unix386.s)
2938                            <1> ;PDE_A_PRESENT equ 1            ; Present flag for PDE
2939                            <1> ;PDE_A_WRITE equ    2           ; Writable (write permission) flag
2940                            <1> ;PDE_A_USER  equ    4           ; User (non-system/kernel) page flag
2941                            <1> ;;
2942                            <1> ;PTE_A_PRESENT equ 1            ; Present flag for PTE (bit 0)
2943                            <1> ;PTE_A_WRITE equ    2           ; Writable (write permission) flag (bit 1)
2944                            <1> ;PTE_A_USER  equ    4           ; User (non-system/kernel) page flag (bit 2)
2945                            <1> ;PTE_A_ACCESS equ  32           ; Accessed flag (bit 5) ; 09/03/2015
2946                            <1>
2947                            <1> ; 27/04/2015
2948                            <1> ; 09/03/2015
2949                            <1> PAGE_SIZE    equ 4096           ; page size in bytes
2950                            <1> PAGE_SHIFT   equ 12             ; page table shift count
2951                            <1> PAGE_D_SHIFT equ 22 ; 12 + 10   ; page directory shift count
2952                            <1> PAGE_OFF     equ 0FFFh          ; 12 bit byte offset in page frame
2953                            <1> PTE_MASK     equ 03FFh          ; page table entry mask
2954                            <1> PTE_DUPLICATED equ 200h         ; duplicated page sign (AVL bit 0)
2955                            <1> PDE_A_CLEAR  equ 0F000h         ; to clear PDE attribute bits
2956                            <1> PTE_A_CLEAR  equ 0F000h         ; to clear PTE attribute bits
2957                            <1> LOGIC_SECT_SIZE equ 512         ; logical sector size
2958                            <1> ERR_MAJOR_PF equ 0E0h           ; major error: page fault
2959                            <1> ERR_MINOR_IM equ 4 ;15/10/2016 (1->4); insufficient (out of) memory
2960                            <1> ERR_MINOR_PV equ 6 ;15/10/2016 (1->4); protection violation
2961                            <1> SWP_DISK_READ_ERR     equ 40
2962                            <1> SWP_DISK_NOT_PRESENT_ERR   equ 41
2963                            <1> SWP_SECTOR_NOT_PRESENT_ERR equ 42
2964                            <1> SWP_NO_FREE_SPACE_ERR      equ 43
2965                            <1> SWP_DISK_WRITE_ERR         equ 44
2966                            <1> SWP_NO_PAGE_TO_SWAP_ERR    equ 45
2967                            <1> PTE_A_ACCESS_BIT equ 5  ; Bit 5 (accessed flag)
2968                            <1> SECTOR_SHIFT  equ 3      ; sector shift (to convert page block number)
2969                            <1> ; 12/07/2016
2970                            <1> PTE_SHARED   equ 400h           ; AVL bit 1, direct memory access bit
2971                            <1>                                 ; (Indicates that the page is not allocated
2972                            <1>                                 ; for the process, it is a shared or system
2973                            <1>                                       ; page, it must not be deallocated!)
2974                            <1> ; 14/12/2020
2975                            <1> ; (Linear Frame Buffer - video memory mark : AVL bit 1, outside M.A.T.)
2976                            <1> PDE_EXTERNAL equ 400h     ; Page directory entry for external memory blocks
2977                            <1> PTE_EXTERNAL equ 400h     ; Allocated kernel pages for Linear Frame Buffer
2978                            <1>                           ; (Out of memory allocation table)
2979                            <1>
2980                            <1> ;
2981                            <1> ;; Retro Unix 386 v1 - paging method/principles
2982                            <1> ;;
2983                            <1> ;; 10/10/2014
2984                            <1> ;; RETRO UNIX 386 v1 - PAGING METHOD/PRINCIPLES
2985                            <1> ;;
2986                            <1> ;; KERNEL PAGE MAP: 1 to 1 physical memory page map
2987                            <1> ;;    (virtual address = physical address)
2988                            <1> ;; KERNEL PAGE TABLES:
2989                            <1> ;;    Kernel page directory and all page tables are
2990                            <1> ;;    on memory as initialized, as equal to physical memory
2991                            <1> ;;    layout. Kernel pages can/must not be swapped out/in.
2992                            <1> ;;
2993                            <1> ;;    what for: User pages may be swapped out, when accessing
2994                            <1> ;;    a page in kernel/system mode, if it would be swapped out,
2995                            <1> ;;    kernel would have to swap it in! But it is also may be
2996                            <1> ;;    in use by a user process. (In system/kernel mode
2997                            <1> ;;    kernel can access all memory pages even if they are
2998                            <1> ;;    reserved/allocated for user processes. Swap out/in would
2999                            <1> ;;    cause conflicts.)
3000                            <1> ;;
3001                            <1> ;;    As result of these conditions,
3002                            <1> ;;    all kernel pages must be initialized as equal to
3003                            <1> ;;    physical layout for preventing page faults.
3004                            <1> ;;    Also, calling "allocate page" procedure after
3005                            <1> ;;    a page fault can cause another page fault (double fault)
3006                            <1> ;;    if all kernel page tables would not be initialized.
3007                            <1> ;;
3008                            <1> ;;    [first_page] = Beginning of users space, as offset to
3009                            <1> ;;    memory allocation table. (double word aligned)
3010                            <1> ;;
3011                            <1> ;;    [next_page] = first/next free space to be searched
3012                            <1> ;;    as offset to memory allocation table. (dw aligned)
3013                            <1> ;;
```

```
3014                    <1> ;;    [last_page] = End of memory (users space), as offset
3015                    <1> ;;    to memory allocation table. (double word aligned)
3016                    <1> ;;
3017                    <1> ;; USER PAGE TABLES:
3018                    <1> ;;    Demand paging (& 'copy on write' allocation method) ...
3019                    <1> ;;           'ready only' marked copies of the
3020                    <1> ;;           parent process's page table entries (for
3021                    <1> ;;           same physical memory).
3022                    <1> ;;           (A page will be copied to a new page after
3023                    <1> ;;            if it causes R/W page fault.)
3024                    <1> ;;
3025                    <1> ;;    Every user process has own (different)
3026                    <1> ;;    page directory and page tables.
3027                    <1> ;;
3028                    <1> ;;    Code starts at virtual address 0, always.
3029                    <1> ;;    (Initial value of EIP is 0 in user mode.)
3030                    <1> ;;    (Programs can be written/developed as simple
3031                    <1> ;;     flat memory programs.)
3032                    <1> ;;
3033                    <1> ;; MEMORY ALLOCATION STRATEGY:
3034                    <1> ;;    Memory page will be allocated by kernel only
3035                    <1> ;;           (in kernel/system mode only).
3036                    <1> ;;    * After a
3037                    <1> ;;      - 'not present' page fault
3038                    <1> ;;      - 'writing attempt on read only page' page fault
3039                    <1> ;;    * For loading (opening, reading) a file or disk/drive
3040                    <1> ;;    * As responce to 'allocate additional memory blocks'
3041                    <1> ;;      request by running process.
3042                    <1> ;;    * While creating a process, allocating a new buffer,
3043                    <1> ;;      new page tables etc.
3044                    <1> ;;
3045                    <1> ;;    At first,
3046                    <1> ;;    - 'allocate page' procedure will be called;
3047                    <1> ;,       if it will return with a valid (>0) physical address
3048                    <1> ;;       (that means the relevant M.A.T. bit has been RESET)
3049                    <1> ;;       relevant memory page/block will be cleared (zeroed).
3050                    <1> ;;    - 'allocate page' will be called for allocating page
3051                    <1> ;;       directory, page table and running space (data/code).
3052                    <1> ;;    - every successful 'allocate page' call will decrease
3053                    <1> ;;      'free_pages' count (pointer).
3054                    <1> ;;    - 'out of (insufficient) memory error' will be returned
3055                    <1> ;;      if 'free_pages' points to a ZERO.
3056                    <1> ;;    - swapping out and swapping in (if it is not a new page)
3057                    <1> ;;      procedures will be called as responce to 'out of memory'
3058                    <1> ;;      error except errors caused by attribute conflicts.
3059                    <1> ;;     (swapper functions)
3060                    <1> ;;
3061                    <1> ;;    At second,
3062                    <1> ;;    - page directory entry will be updated then page table
3063                    <1> ;;      entry will be updated.
3064                    <1> ;;
3065                    <1> ;; MEMORY ALLOCATION TABLE FORMAT:
3066                    <1> ;;    - M.A.T. has a size according to available memory as
3067                    <1> ;;      follows:
3068                    <1> ;;            - 1 (allocation) bit per 1 page (4096 bytes)
3069                    <1> ;;            - a bit with value of 0 means allocated page
3070                    <1> ;;            - a bit with value of 1 means a free page
3071                    <1> ;,    - 'free_pages' pointer holds count of free pages
3072                    <1> ;;      depending on M.A.T.
3073                    <1> ;;           (NOTE: Free page count will not be checked
3074                    <1> ;;           again -on M.A.T.- after initialization.
3075                    <1> ;;           Kernel will trust on initial count.)
3076                    <1> ;,    - 'free_pages' count will be decreased by allocation
3077                    <1> ;;      and it will be increased by deallocation procedures.
3078                    <1> ;;
3079                    <1> ;;    - Available memory will be calculated during
3080                    <1> ;;      the kernel's initialization stage (in real mode).
3081                    <1> ;;      Memory allocation table and kernel page tables
3082                    <1> ;;      will be formatted/sized as result of available
3083                    <1> ;;      memory calculation before paging is enabled.
3084                    <1> ;;
3085                    <1> ;; For 4GB Available/Present Memory: (max. possible memory size)
3086                    <1> ;;    - Memory Allocation Table size will be 128 KB.
3087                    <1> ;;    - Memory allocation for kernel page directory size
3088                    <1> ;;      is always 4 KB. (in addition to total allocation size
3089                    <1> ;;      for page tables)
3090                    <1> ;;    - Memory allocation for kernel page tables (1024 tables)
3091                    <1> ;;      is 4 MB (1024*4*1024 bytes).
3092                    <1> ;;    - User (available) space will be started
3093                    <1> ;;      at 6th MB of the memory (after 1MB+4MB).
3094                    <1> ;;    - The first 640 KB is for kernel's itself plus
3095                    <1> ;;      memory allocation table and kernel's page directory
3096                    <1> ;;      (D0000h-EFFFFh may be used as kernel space...)
3097                    <1> ;;    - B0000h to B7FFFh address space (32 KB) will be used
3098                    <1> ;;      for buffers.
3099                    <1> ;;    - ROMBIOS, VIDEO BUFFER and VIDEO ROM space are reserved.
3100                    <1> ;,      (A0000h-AFFFFh, C0000h-CFFFFh, F0000h-FFFFFh)
3101                    <1> ;;    - Kernel page tables start at 100000h (2nd MB)
3102                    <1> ;;
3103                    <1> ;; For 1GB Available Memory:
3104                    <1> ;;    - Memory Allocation Table size will be 32 KB.
3105                    <1> ;;    - Memory allocation for kernel page directory size
3106                    <1> ;;      is always 4 KB. (in addition to total allocation size
3107                    <1> ;;      for page tables)
3108                    <1> ;;    - Memory allocation for kernel page tables (256 tables)
3109                    <1> ;;      is 1 MB (256*4*1024 bytes).
3110                    <1> ;;    - User (available) space will be started
3111                    <1> ;;      at 3th MB of the memory (after 1MB+1MB).
3112                    <1> ;;    - The first 640 KB is for kernel's itself plus
3113                    <1> ;;      memory allocation table and kernel's page directory
3114                    <1> ;;      (D0000h-EFFFFh may be used as kernel space...)
3115                    <1> ;;    - B0000h to B7FFFh address space (32 KB) will be used
3116                    <1> ;;      for buffers.
3117                    <1> ;;    - ROMBIOS, VIDEO BUFFER and VIDEO ROM space are reserved.
3118                    <1> ;,      (A0000h-AFFFFh, C0000h-CFFFFh, F0000h-FFFFFh)
```

```
3119                    <1> ;;    - Kernel page tables start at 100000h (2nd MB).
3120                    <1> ;;
3121                    <1> ;;
3122                    <1>
3123                    <1>
3124                    <1> ;;////////////////////////////////////////////////////////////////////////////
3125                    <1> ;;
3126                    <1> ;; RETRO UNIX 386 v1 - Paging (Method for Copy On Write paging principle)
3127                    <1> ;; DEMAND PAGING - PARENT&CHILD PAGE TABLE DUPLICATION PRINCIPLES (23/04/2015)
3128                    <1>
3129                    <1> ;; Main factor: "sys fork" system call
3130                    <1> ;;
3131                    <1> ;;          FORK
3132                    <1> ;;                      |----> parent - duplicated PTEs, read only pages
3133                    <1> ;;  writable pages ---->|
3134                    <1> ;;                      |----> child - duplicated PTEs, read only pages
3135                    <1> ;;
3136                    <1> ;; AVL bit (0) of Page Table Entry is used as duplication sign
3137                    <1> ;;
3138                    <1> ;; AVL Bit 0 [PTE Bit 9] = 'Duplicated PTE belongs to child' sign/flag (if it is set)
3139                    <1> ;; Note: Dirty bit (PTE bit 6) may be used instead of AVL bit 0 (PTE bit 9)
3140                    <1> ;;       -while R/W bit is 0-.
3141                    <1> ;;
3142                    <1> ;; Duplicate page tables with writable pages (the 1st sys fork in the process):
3143                    <1> ;; # Parent's Page Table Entries are updated to point same pages as read only,
3144                    <1> ;;   as duplicated PTE bit  -AVL bit 0, PTE bit 9- are reset/clear.
3145                    <1> ;; # Then Parent's Page Table is copied to Child's Page Table.
3146                    <1> ;; # Child's Page Table Entries are updated as duplicated child bit
3147                    <1> ;;   -AVL bit 0, PTE bit 9- is set.
3148                    <1> ;;
3149                    <1> ;; Duplicate page tables with read only pages (several sys fork system calls):
3150                    <1> ;; # Parent's read only pages are copied to new child pages.
3151                    <1> ;;   Parent's PTE attributes are not changed.
3152                    <1> ;;   (Because, there is another parent-child fork before this fork! We must not
3153                    <1> ;;    destroy/mix previous fork result).
3154                    <1> ;; # Child's Page Table Entries (which are corresponding to Parent's
3155                    <1> ;;   read only pages) are set as writable (while duplicated PTE bit is clear).
3156                    <1> ;; # Parent's PTEs with writable page attribute are updated to point same pages
3157                    <1> ;;   as read only, (while) duplicated PTE bit is reset (clear).
3158                    <1> ;; # Parent's Page Table Entries (with writable page attribute) are duplicated
3159                    <1> ;;   as Child's Page Table Entries without copying actual page.
3160                    <1> ;; # Child 's Page Table Entries (which are corresponding to Parent's writable
3161                    <1> ;;   pages) are updated as duplicated PTE bit (AVL bit 0, PTE bit 9- is set.
3162                    <1> ;;
3163                    <1> ;; !? WHAT FOR (duplication after duplication):
3164                    <1> ;; In UNIX method for sys fork (a typical 'fork' application in /etc/init)
3165                    <1> ;; program/executable code continues from specified location as child process,
3166                    <1> ;; returns back previous code location as parent process, every child after
3167                    <1> ;; every sys fork uses last image of code and data just prior the fork.
3168                    <1> ;; Even if the parent code changes data, the child will not see the changed data
3169                    <1> ;; after the fork. In Retro UNIX 8086 v1, parent's process segment (32KB)
3170                    <1> ;; was copied to child's process segment (all of code and data) according to
3171                    <1> ;; original UNIX v1 which copies all of parent process code and data -core-
3172                    <1> ;; to child space -core- but swaps that core image -of child- on to disk.
3173                    <1> ;; If I (Erdogan Tan) would use a method of to copy parent's core
3174                    <1> ;; (complete running image of parent process) to the child process;
3175                    <1> ;; for big sizes, i would force Retro UNIX 386 v1 to spend many memory pages
3176                    <1> ;; and times only for a sys fork. (It would excessive reservation for sys fork,
3177                    <1> ;; because sys fork usually is prior to sys exec; sys exec always establishes
3178                    <1> ;; a new/fresh core -running space-, by clearing all code/data content).
3179                    <1> ;; 'Read Only' page flag ensures page fault handler is needed only for a few write
3180                    <1> ;; attempts between sys fork and sys exec, not more... (I say so by thinking
3181                    <1> ;; of "/etc/init" content, specially.) sys exec will clear page tables and
3182                    <1> ;; new/fresh pages will be used to load and run new executable/program.
3183                    <1> ;; That is what for i have preferred "copy on write", "duplication" method
3184                    <1> ;; for sharing same read only pages between parent and child processes.
3185                    <1> ;; That is a pitty i have to use new private flag (AVL bit 0, "duplicated PTE
3186                    <1> ;; belongs to child" sign) for cooperation on duplicated pages between a parent
3187                    <1> ;; and it's child processes; otherwise parent process would destroy data belongs
3188                    <1> ;; to its child or vice versa; or some pages would remain unclaimed
3189                    <1> ;; -deallocation problem-.
3190                    <1> ;; Note: to prevent conflicts, read only pages must not be swapped out...
3191                    <1> ;;
3192                    <1> ;; WHEN PARENT TRIES TO WRITE IT'S READ ONLY (DUPLICATED) PAGE:
3193                    <1> ;; # Page fault handler will do those:
3194                    <1> ;;   - 'Duplicated PTE' flag (PTE bit 9) is checked (on the failed PTE).
3195                    <1> ;;   - If it is reset/clear, there is a child uses same page.
3196                    <1> ;;   - Parent's read only page -previous page- is copied to a new writable page.
3197                    <1> ;;   - Parent's PTE is updated as writable page, as unique page (AVL=0)
3198                    <1> ;;   - (Page fault handler whill check this PTE later, if child process causes to
3199                    <1> ;;     page fault due to write attempt on read only page. Of course, the previous
3200                    <1> ;;     read only page will be converted to writable and unique page which belongs
3201                    <1> ;;     to child process.)
3202                    <1> ;; WHEN CHILD TRIES TO WRITE IT'S READ ONLY (DUPLICATED) PAGE:
3203                    <1> ;; # Page fault handler will do those:
3204                    <1> ;;   - 'Duplicated PTE' flag (PTE bit 9) is checked (on the failed PTE).
3205                    <1> ;;   - If it is set, there is a parent uses -or was using- same page.
3206                    <1> ;;   - Same PTE address within parent's page table is checked if it has same page
3207                    <1> ;;     address or not.
3208                    <1> ;;   - If parent's PTE has same address, child will continue with a new writable page.
3209                    <1> ;;     Parent's PTE will point to same (previous) page as writable, unique (AVL=0).
3210                    <1> ;;   - If parent's PTE has different address, child will continue with it's
3211                    <1> ;;     own/same page but read only flag (0) will be changed to writable flag (1) and
3212                    <1> ;;     'duplicated PTE (belongs to child)' flag/sign will be cleared/reset.
3213                    <1> ;;
3214                    <1> ;; NOTE: When a child process is terminated, read only flags of parent's page tables
3215                    <1> ;;       will be set as writable (and unique) in case of child process was using
3216                    <1> ;;       same pages with duplicated child PTE sign... Depending on sys fork and
3217                    <1> ;;       duplication method details, it is not possible multiple child processes
3218                    <1> ;;       were using same page with duplicated PTEs.
3219                    <1> ;;
3220                    <1> ;;////////////////////////////////////////////////////////////////////////////
3221                    <1>
3222                    <1> ;; 08/10/2014
3223                    <1> ;; 11/09/2014 - Retro UNIX 386 v1 PAGING (further) draft
```

```
                        <1> ;;          by Erdogan Tan (Based on KolibriOS 'memory.inc')
                        <1>
                        <1> ;; 'allocate_page' code is derived and modified from KolibriOS
                        <1> ;; 'alloc_page' procedure in 'memory.inc'
                        <1> ;; (25/08/2014, Revision: 5057) file
                        <1> ;; by KolibriOS Team (2004-2012)
                        <1>
                        <1> allocate_page:
                        <1>       ; 17/04/2021 - TRDOS 386 v2.0.4
                        <1>       ;       (temporary modifications)
                        <1>       ; 01/07/2015
                        <1>       ; 05/05/2015
                        <1>       ; 30/04/2015
                        <1>       ; 16/10/2014
                        <1>       ; 08/10/2014
                        <1>       ; 09/09/2014 (Retro UNIX 386 v1 - beginning)
                        <1>       ;
                        <1>       ; INPUT -> none
                        <1>       ;
                        <1>       ; OUTPUT ->
                        <1>       ;     EAX = PHYSICAL (real/flat) ADDRESS OF THE ALLOCATED PAGE
                        <1>       ;     (corresponding MEMORY ALLOCATION TABLE bit is RESET)
                        <1>       ;
                        <1>       ;     CF = 1 and EAX = 0
                        <1>       ;               if there is not a free page to be allocated
                        <1>       ;
                        <1>       ; Modified Registers -> none (except EAX)
                        <1>       ;
00005A75 A1[B0800100]   <1>       mov   eax, [free_pages]
00005A7A 21C0           <1>       and   eax, eax
00005A7C 7438           <1>       jz    short out_of_memory
                        <1>       ;
00005A7E 53             <1>       push  ebx
00005A7F 51             <1>       push  ecx
                        <1>       ;
00005A80 BB00001000     <1>       mov   ebx, MEM_ALLOC_TBL   ; Memory Allocation Table offset
00005A85 89D9           <1>       mov   ecx, ebx
                        <1>                                  ; NOTE: 32 (first_page) is initial
                        <1>                                  ; value of [next_page].
                        <1>                                  ; It points to the first available
                        <1>                                  ; page block for users (ring 3) ...
                        <1>                                  ; (MAT offset 32 = 1024/32)
                        <1>                                  ; (at the of the first 4 MB)
00005A87 031D[B4800100] <1>       add   ebx, [next_page] ; Free page searching starts from here
                        <1>                              ; next_free_page >> 5
00005A8D 030D[B8800100] <1>       add   ecx, [last_page] ; Free page searching ends here
                        <1>                              ; (total_pages - 1) >> 5
                        <1> al_p_scan:
00005A93 39CB           <1>       cmp   ebx, ecx
00005A95 770A           <1>       ja    short al_p_notfound
                        <1>       ;
                        <1>       ; 01/07/2015
                        <1>       ; AMD64 Architecture Programmer's Manual
                        <1>       ; Volume 3:
                        <1>       ; General-Purpose and System Instructions
                        <1>       ;
                        <1>       ; BSF - Bit Scan Forward
                        <1>       ;
                        <1>       ;   Searches the value in a register or a memory location
                        <1>       ;   (second operand) for the least-significant set bit.
                        <1>       ;   If a set bit is found, the instruction clears the zero flag (ZF)
                        <1>       ;   and stores the index of the least-significant set bit in a destination
                        <1>       ;   register (first operand). If the second operand contains 0,
                        <1>       ;   the instruction sets ZF to 1 and does not change the contents of the
                        <1>       ;   destination register. The bit index is an unsigned offset from bit 0
                        <1>       ;   of the searched value
                        <1>       ;
00005A97 0FBC03         <1>       bsf   eax, [ebx] ; Scans source operand for first bit set (1).
                        <1>                        ; Clear ZF if a bit is found set (1) and
                        <1>                        ; loads the destination with an index to
                        <1>                        ; first set bit. (0 -> 31)
                        <1>                        ; Sets ZF to 1 if no bits are found set.
00005A9A 751C           <1>       jnz   short al_p_found ; ZF = 0 -> a free page has been found
                        <1>                        ;
                        <1>                        ; NOTE:  a Memory Allocation Table bit
                        <1>                        ;       with value of 1 means
                        <1>                        ;       the corresponding page is free
                        <1>                        ;       (Retro UNIX 386 v1 feature only!)
00005A9C 83C304         <1>       add   ebx, 4
                        <1>                        ; We return back for searching next page block
                        <1>                        ; NOTE: [free_pages] is not ZERO; so,
                        <1>                        ;     we always will find at least 1 free page here.
00005A9F EBF2           <1>       jmp   short al_p_scan
                        <1>       ;
                        <1> al_p_notfound:
00005AA1 81E900001000   <1>       sub   ecx, MEM_ALLOC_TBL
00005AA7 890D[B4800100] <1>       mov   [next_page], ecx ; next/first free page = last page
                        <1>                              ; (deallocate_page procedure will change it)
00005AAD 31C0           <1>       xor   eax, eax
00005AAF A3[B0800100]   <1>       mov   [free_pages], eax ; 0
00005AB4 59             <1>       pop   ecx
00005AB5 5B             <1>       pop   ebx
                        <1>       ;
                        <1> ; 17/04/2021
                        <1> ; ('swap_out' procedure call is disabled as temporary)
                        <1>
                        <1> out_of_memory:
                        <1> ;     call  swap_out
                        <1> ;     jnc   short al_p_ok  ; [free_pages] = 0, re-allocation by swap_out
                        <1> ;     ;
                        <1> ;     sub   eax, eax ; 0
00005AB6 F9             <1>       stc
00005AB7 C3             <1>       retn
                        <1>
                        <1> al_p_found:
```

```
3329 00005AB8 89D9                <1>       mov    ecx, ebx
3330 00005ABA 81E900001000        <1>       sub    ecx, MEM_ALLOC_TBL
3331 00005AC0 890D[B4800100]      <1>       mov    [next_page], ecx ; Set first free page searching start
3332                              <1>                               ; address/offset (to the next)
3333 00005AC6 FF0D[B0800100]      <1>       dec   dword [free_pages] ; 1 page has been allocated (X = X-1)
3334                              <1>       ;
3335 00005ACC 0FB303              <1>       btr    [ebx], eax     ; The destination bit indexed by the source value
3336                              <1>                               ; is copied into the Carry Flag and then cleared
3337                              <1>                               ; in the destination.
3338                              <1>                               ;
3339                              <1>                               ; Reset the bit which is corresponding to the
3340                              <1>                               ; (just) allocated page.
3341                              <1>       ; 01/07/2015 (4*8 = 32, 1 allocation byte = 8 pages)
3342 00005ACF C1E103              <1>       shl    ecx, 3         ; (page block offset * 32) + page index
3343 00005AD2 01C8                <1>       add    eax, ecx       ; = page number
3344 00005AD4 C1E00C              <1>       shl    eax, 12              ; physical address of the page (flat/real value)
3345                              <1>       ; EAX = physical address of memory page
3346                              <1>       ;
3347                              <1>       ; NOTE: The relevant page directory and page table entry will be updated
3348                              <1>       ;       according to this EAX value...
3349 00005AD7 59                  <1>       pop    ecx
3350 00005AD8 5B                  <1>       pop    ebx
3351                              <1> al_p_ok:
3352 00005AD9 C3                  <1>       retn
3353                              <1>
3354                              <1> make_page_dir:
3355                              <1>       ; 18/04/2015
3356                              <1>       ; 12/04/2015
3357                              <1>       ; 23/10/2014
3358                              <1>       ; 16/10/2014
3359                              <1>       ; 09/10/2014 ; (Retro UNIX 386 v1 - beginning)
3360                              <1>       ;
3361                              <1>       ; INPUT ->
3362                              <1>       ;      none
3363                              <1>       ; OUTPUT ->
3364                              <1>       ;      (EAX = 0)
3365                              <1>       ;      cf = 1 -> insufficient (out of) memory error
3366                              <1>       ;      cf = 0 ->
3367                              <1>       ;      u.pgdir = page directory (physical) address of the current
3368                              <1>       ;               process/user.
3369                              <1>       ;
3370                              <1>       ; Modified Registers -> EAX
3371                              <1>       ;
3372 00005ADA E896FFFFFF          <1>       call   allocate_page
3373 00005ADF 7216                <1>       jc     short mkpd_error
3374                              <1>       ;
3375 00005AE1 A3[B8030300]        <1>       mov    [u.pgdir], eax    ; Page dir address for current user/process
3376                              <1>                                ; (Physical address)
3377                              <1> clear_page:
3378                              <1>       ; 18/04/2015
3379                              <1>       ; 09/10/2014 ; (Retro UNIX 386 v1 - beginning)
3380                              <1>       ;
3381                              <1>       ; INPUT ->
3382                              <1>       ;      EAX = physical address of the page
3383                              <1>       ; OUTPUT ->
3384                              <1>       ;      all bytes of the page will be cleared
3385                              <1>       ;
3386                              <1>       ; Modified Registers -> none
3387                              <1>       ;
3388 00005AE6 57                  <1>       push   edi
3389 00005AE7 51                  <1>       push   ecx
3390 00005AE8 50                  <1>       push   eax
3391 00005AE9 B900040000          <1>       mov    ecx, PAGE_SIZE / 4
3392 00005AEE 89C7                <1>       mov    edi, eax
3393 00005AF0 31C0                <1>       xor    eax, eax
3394 00005AF2 F3AB                <1>       rep    stosd
3395 00005AF4 58                  <1>       pop    eax
3396 00005AF5 59                  <1>       pop    ecx
3397 00005AF6 5F                  <1>       pop    edi
3398                              <1> mkpd_error:
3399                              <1> mkpt_error:
3400 00005AF7 C3                  <1>       retn
3401                              <1>
3402                              <1> make_page_table:
3403                              <1>       ; 23/06/2015
3404                              <1>       ; 18/04/2015
3405                              <1>       ; 12/04/2015
3406                              <1>       ; 16/10/2014
3407                              <1>       ; 09/10/2014 ; (Retro UNIX 386 v1 - beginning)
3408                              <1>       ;
3409                              <1>       ; INPUT ->
3410                              <1>       ;      EBX = virtual (linear) address
3411                              <1>       ;      ECX = page table attributes (lower 12 bits)
3412                              <1>       ;            (higher 20 bits must be ZERO)
3413                              <1>       ;            (bit 0 must be 1)
3414                              <1>       ;      u.pgdir = page directory (physical) address
3415                              <1>       ; OUTPUT ->
3416                              <1>       ;      EDX = Page directory entry address
3417                              <1>       ;      EAX = Page table address
3418                              <1>       ;      cf = 1 -> insufficient (out of) memory error
3419                              <1>       ;      cf = 0 -> page table address in the PDE (EDX)
3420                              <1>       ;
3421                              <1>       ; Modified Registers -> EAX, EDX
3422                              <1>       ;
3423 00005AF8 E878FFFFFF          <1>       call   allocate_page
3424 00005AFD 72F8                <1>       jc     short mkpt_error
3425 00005AFF E811000000          <1>       call   set_pde
3426 00005B04 EBE0                <1>       jmp    short clear_page
3427                              <1>
3428                              <1> make_page:
3429                              <1>       ; 24/07/2015
3430                              <1>       ; 23/06/2015 ; (Retro UNIX 386 v1 - beginning)
3431                              <1>       ;
3432                              <1>       ; INPUT ->
3433                              <1>       ;      EBX = virtual (linear) address
```

```
                                <1>       ;      ECX = page attributes (lower 12 bits)
                                <1>       ;            (higher 20 bits must be ZERO)
                                <1>       ;            (bit 0 must be 1)
                                <1>       ;      u.pgdir = page directory (physical) address
                                <1>       ; OUTPUT ->
                                <1>       ;      EBX = Virtual address
                                <1>       ;      (EDX = PTE value)
                                <1>       ;      EAX = Physical address
                                <1>       ;      cf = 1 -> insufficient (out of) memory error
                                <1>       ;
                                <1>       ; Modified Registers -> EAX, EDX
                                <1>       ;
00005B06 E86AFFFFFF             <1>       call   allocate_page
00005B0B 7207                   <1>       jc     short mkp_err
00005B0D E821000000             <1>       call   set_pte
00005B12 73D2                   <1>       jnc    short clear_page ; 18/04/2015
                                <1> mkp_err:
00005B14 C3                     <1>       retn
                                <1>
                                <1>
                                <1> set_pde:    ; Set page directory entry (PDE)
                                <1>       ; 20/07/2015
                                <1>       ; 18/04/2015
                                <1>       ; 12/04/2015
                                <1>       ; 23/10/2014
                                <1>       ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;      EAX = physical address
                                <1>       ;            (use present value if EAX = 0)
                                <1>       ;      EBX = virtual (linear) address
                                <1>       ;      ECX = page table attributes (lower 12 bits)
                                <1>       ;            (higher 20 bits must be ZERO)
                                <1>       ;            (bit 0 must be 1)
                                <1>       ;      u.pgdir = page directory (physical) address
                                <1>       ; OUTPUT ->
                                <1>       ;      EDX = PDE address
                                <1>       ;      EAX = page table address (physical)
                                <1>       ;      ;(CF=1 -> Invalid page address)
                                <1>       ;
                                <1>       ; Modified Registers -> EDX
                                <1>       ;
00005B15 89DA                   <1>       mov    edx, ebx
00005B17 C1EA16                 <1>       shr    edx, PAGE_D_SHIFT ; 22
00005B1A C1E202                 <1>       shl    edx, 2 ; offset to page directory (1024*4)
00005B1D 0315[B8030300]         <1>       add    edx, [u.pgdir]
                                <1>       ;
00005B23 21C0                   <1>       and    eax, eax
00005B25 7506                   <1>       jnz    short spde_1
                                <1>       ;
00005B27 8B02                   <1>       mov    eax, [edx]  ; old PDE value
                                <1>       ;test  al, 1
                                <1>       ;jz    short spde_2
00005B29 662500F0               <1>       and    ax, PDE_A_CLEAR ; 0F000h  ; clear lower 12 bits
                                <1> spde_1:
                                <1>       ;and   cx, 0FFFh
00005B2D 8902                   <1>       mov    [edx], eax
00005B2F 66090A                 <1>       or     [edx], cx
00005B32 C3                     <1>       retn
                                <1> ;spde_2: ; error
                                <1> ;     stc
                                <1> ;     retn
                                <1>
                                <1> set_pte:    ; Set page table entry (PTE)
                                <1>       ; 24/07/2015
                                <1>       ; 20/07/2015
                                <1>       ; 23/06/2015
                                <1>       ; 18/04/2015
                                <1>       ; 12/04/2015
                                <1>       ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;      EAX = physical page address
                                <1>       ;            (use present value if EAX = 0)
                                <1>       ;      EBX = virtual (linear) address
                                <1>       ;      ECX = page attributes (lower 12 bits)
                                <1>       ;            (higher 20 bits must be ZERO)
                                <1>       ;            (bit 0 must be 1)
                                <1>       ;      u.pgdir = page directory (physical) address
                                <1>       ; OUTPUT ->
                                <1>       ;      EAX = physical page address
                                <1>       ;      (EDX = PTE value)
                                <1>       ;      EBX = virtual address
                                <1>       ;
                                <1>       ;      CF = 1 -> error
                                <1>       ;
                                <1>       ; Modified Registers -> EAX, EDX
                                <1>       ;
00005B33 50                     <1>       push   eax
00005B34 A1[B8030300]           <1>       mov    eax, [u.pgdir] ; 20/07/2015
00005B39 E837000000             <1>       call   get_pde
                                <1>              ; EDX = PDE address
                                <1>              ; EAX = PDE value
00005B3E 5A                     <1>       pop    edx ; physical page address
00005B3F 722A                   <1>       jc     short spte_err ; PDE not present
                                <1>       ;
00005B41 53                     <1>       push   ebx ; 24/07/2015
00005B42 662500F0               <1>       and    ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 bits
                                <1>                        ; EDX = PT address (physical)
00005B46 C1EB0C                 <1>       shr    ebx, PAGE_SHIFT ; 12
00005B49 81E3FF030000           <1>       and    ebx, PTE_MASK; 03FFh
                                <1>                     ; clear higher 10 bits (PD bits)
00005B4F C1E302                 <1>       shl    ebx, 2   ; offset to page table (1024*4)
00005B52 01C3                   <1>       add    ebx, eax
                                <1>       ;
```

```
3539 00005B54 8B03              <1>       mov   eax, [ebx] ; Old PTE value
3540 00005B56 A801              <1>       test  al, 1
3541 00005B58 740C              <1>       jz    short spte_0
3542 00005B5A 09D2              <1>       or    edx, edx
3543 00005B5C 750F              <1>       jnz   short spte_1
3544 00005B5E 662500F0          <1>       and   ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 bits
3545 00005B62 89C2              <1>       mov   edx, eax
3546 00005B64 EB09              <1>       jmp   short spte_2
3547                            <1> spte_0:
3548                            <1>       ; If this PTE contains a swap (disk) address,
3549                            <1>       ; it can be updated by using 'swap_in' procedure
3550                            <1>       ; only!
3551 00005B66 21C0              <1>       and   eax, eax
3552 00005B68 7403              <1>       jz    short spte_1
3553                            <1>       ; 24/07/2015
3554                            <1>       ; swapped page ! (on disk)
3555 00005B6A 5B                <1>       pop   ebx
3556                            <1> spte_err:
3557 00005B6B F9                <1>       stc
3558 00005B6C C3                <1>       retn
3559                            <1> spte_1:
3560 00005B6D 89D0              <1>       mov   eax, edx
3561                            <1> spte_2:
3562 00005B6F 09CA              <1>       or    edx, ecx
3563                            <1>       ; 23/06/2015
3564 00005B71 8913              <1>       mov   [ebx], edx ; PTE value in EDX
3565                            <1>       ; 24/07/2015
3566 00005B73 5B                <1>       pop   ebx
3567 00005B74 C3                <1>       retn
3568                            <1>
3569                            <1> get_pde:    ; Get present value of the relevant PDE
3570                            <1>       ; 20/07/2015
3571                            <1>       ; 18/04/2015
3572                            <1>       ; 12/04/2015
3573                            <1>       ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
3574                            <1>       ;
3575                            <1>       ; INPUT ->
3576                            <1>       ;     EBX = virtual (linear) address
3577                            <1>       ;     EAX = page directory (physical) address
3578                            <1>       ; OUTPUT ->
3579                            <1>       ;     EDX = Page directory entry address
3580                            <1>       ;     EAX = Page directory entry value
3581                            <1>       ;     CF = 1 -> PDE not present or invalid ?
3582                            <1>       ; Modified Registers -> EDX, EAX
3583                            <1>       ;
3584 00005B75 89DA              <1>       mov   edx, ebx
3585 00005B77 C1EA16            <1>       shr   edx, PAGE_D_SHIFT ; 22  (12+10)
3586 00005B7A C1E202            <1>       shl   edx, 2 ; offset to page directory (1024*4)
3587 00005B7D 01C2              <1>       add   edx, eax ; page directory address (physical)
3588 00005B7F 8B02              <1>       mov   eax, [edx]
3589 00005B81 A801              <1>       test  al, PDE_A_PRESENT ; page table is present or not !
3590 00005B83 751F              <1>       jnz   short gpte_retn
3591 00005B85 F9                <1>       stc
3592                            <1> gpde_retn:
3593 00005B86 C3                <1>       retn
3594                            <1>
3595                            <1> get_pte:
3596                            <1>             ; Get present value of the relevant PTE
3597                            <1>       ; 29/07/2015
3598                            <1>       ; 20/07/2015
3599                            <1>       ; 18/04/2015
3600                            <1>       ; 12/04/2015
3601                            <1>       ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
3602                            <1>       ;
3603                            <1>       ; INPUT ->
3604                            <1>       ;     EBX = virtual (linear) address
3605                            <1>       ;     EAX = page directory (physical) address
3606                            <1>       ; OUTPUT ->
3607                            <1>       ;     EDX = Page table entry address (if CF=0)
3608                            <1>       ;           Page directory entry address (if CF=1)
3609                            <1>       ;            (Bit 0 value is 0 if PT is not present)
3610                            <1>       ;     EAX = Page table entry value (page address)
3611                            <1>       ;     CF = 1 -> PDE not present or invalid ?
3612                            <1>       ; Modified Registers -> EAX, EDX
3613                            <1>       ;
3614 00005B87 E8E9FFFFFF        <1>       call  get_pde
3615 00005B8C 72F8              <1>       jc    short gpde_retn     ; page table is not present
3616                            <1>       ;jnc  short gpte_1
3617                            <1>       ;retn
3618                            <1> ;gpte_1:
3619 00005B8E 662500F0          <1>       and   ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 bits
3620 00005B92 89DA              <1>       mov   edx, ebx
3621 00005B94 C1EA0C            <1>       shr   edx, PAGE_SHIFT ; 12
3622 00005B97 81E2FF030000      <1>       and   edx, PTE_MASK; 03FFh
3623                            <1>                     ; clear higher 10 bits (PD bits)
3624 00005B9D C1E202            <1>       shl   edx, 2 ; offset from start of page table (1024*4)
3625 00005BA0 01C2              <1>       add   edx, eax
3626 00005BA2 8B02              <1>       mov   eax, [edx]
3627                            <1> gpte_retn:
3628 00005BA4 C3                <1>       retn
3629                            <1>
3630                            <1> deallocate_page_dir:
3631                            <1>       ; 15/09/2015
3632                            <1>       ; 05/08/2015
3633                            <1>       ; 30/04/2015
3634                            <1>       ; 28/04/2015
3635                            <1>       ; 17/10/2014
3636                            <1>       ; 12/10/2014 (Retro UNIX 386 v1 - beginning)
3637                            <1>       ;
3638                            <1>       ; INPUT ->
3639                            <1>       ;     EAX = PHYSICAL ADDRESS OF THE PAGE DIRECTORY (CHILD)
3640                            <1>       ;     EBX = PHYSICAL ADDRESS OF THE PARENT'S PAGE DIRECTORY
3641                            <1>       ; OUTPUT ->
3642                            <1>       ;     All of page tables in the page directory
3643                            <1>       ;     and page dir's itself will be deallocated
```

```
3644                                    <1>       ;      except 'read only' duplicated pages (will be converted
3645                                    <1>       ;      to writable pages).
3646                                    <1>       ;
3647                                    <1>       ; Modified Registers -> EAX
3648                                    <1>       ;
3649                                    <1>       ;
3650 00005BA5 56                       <1>       push   esi
3651 00005BA6 51                       <1>       push   ecx
3652 00005BA7 50                       <1>       push   eax
3653 00005BA8 89C6                     <1>       mov    esi, eax
3654 00005BAA 31C9                     <1>       xor    ecx, ecx
3655                                    <1>       ; The 1st PDE points to Kernel Page Table 0 (the 1st 4MB),
3656                                    <1>       ; it must not be deallocated
3657 00005BAC 890E                     <1>       mov    [esi], ecx ; 0 ; clear PDE 0
3658                                    <1> dapd_0:
3659 00005BAE AD                       <1>       lodsd
3660 00005BAF A801                     <1>       test   al, PDE_A_PRESENT ; bit 0, present flag (must be 1)
3661 00005BB1 7409                     <1>       jz     short dapd_1
3662 00005BB3 662500F0                 <1>       and    ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
3663 00005BB7 E812000000               <1>       call   deallocate_page_table
3664                                    <1> dapd_1:
3665 00005BBC 41                       <1>       inc    ecx ; page directory entry index
3666 00005BBD 81F900040000             <1>       cmp    ecx, PAGE_SIZE / 4 ; 1024
3667 00005BC3 72E9                     <1>       jb     short dapd_0
3668                                    <1> dapd_2:
3669 00005BC5 58                       <1>       pop    eax
3670 00005BC6 E872000000               <1>       call   deallocate_page     ; deallocate the page dir's itself
3671 00005BCB 59                       <1>       pop    ecx
3672 00005BCC 5E                       <1>       pop    esi
3673 00005BCD C3                       <1>       retn
3674                                    <1>
3675                                    <1> deallocate_page_table:
3676                                    <1>       ; 17/04/2021 - TRDOS 386 v2.0.4
3677                                    <1>       ;      (temporary modifications)
3678                                    <1>       ; 12/07/2016
3679                                    <1>       ; 19/09/2015
3680                                    <1>       ; 15/09/2015
3681                                    <1>       ; 05/08/2015
3682                                    <1>       ; 30/04/2015
3683                                    <1>       ; 28/04/2015
3684                                    <1>       ; 24/10/2014
3685                                    <1>       ; 23/10/2014
3686                                    <1>       ; 12/10/2014 (Retro UNIX 386 v1 - beginning)
3687                                    <1>       ;
3688                                    <1>       ; INPUT ->
3689                                    <1>       ;      EAX = PHYSICAL (real/flat) ADDRESS OF THE PAGE TABLE
3690                                    <1>       ;      EBX = PHYSICAL ADDRESS OF THE PARENT'S PAGE DIRECTORY
3691                                    <1>       ;      (ECX = page directory entry index)
3692                                    <1>       ; OUTPUT ->
3693                                    <1>       ;      All of pages in the page table and page table's itself
3694                                    <1>       ;      will be deallocated except 'read only' duplicated pages
3695                                    <1>       ;      (will be converted to writable pages).
3696                                    <1>       ;
3697                                    <1>       ; Modified Registers -> EAX
3698                                    <1>       ;
3699 00005BCE 56                       <1>       push   esi
3700 00005BCF 57                       <1>       push   edi
3701 00005BD0 52                       <1>       push   edx
3702 00005BD1 50                       <1>       push   eax ; *
3703 00005BD2 89C6                     <1>       mov    esi, eax
3704 00005BD4 31FF                     <1>       xor    edi, edi ; 0
3705                                    <1> dapt_0:
3706 00005BD6 AD                       <1>       lodsd
3707 00005BD7 A801                     <1>       test   al, PTE_A_PRESENT ; bit 0, present flag (must be 1)
3708 00005BD9 7455                     <1>       jz     short dapt_1
3709                                    <1>       ;
3710 00005BDB A802                     <1>       test   al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
3711                                    <1>                                ; (must be 1)
3712 00005BDD 753F                     <1>       jnz    short dapt_3
3713                                    <1>       ; Read only -duplicated- page (belongs to a parent or a child)
3714 00005BDF 66A90002                 <1>         test    ax, PTE_DUPLICATED ; Was this page duplicated
3715                                    <1>                                  ; as child's page ?
3716 00005BE3 7444                     <1>       jz     short dapt_4 ; Clear PTE but don't deallocate the page!
3717                                    <1>       ; check the parent's PTE value is read only & same page or not..
3718                                    <1>       ; ECX = page directory entry index (0-1023)
3719 00005BE5 53                       <1>       push   ebx
3720 00005BE6 51                       <1>       push   ecx
3721 00005BE7 66C1E102                 <1>       shl    cx, 2 ; *4
3722 00005BEB 01CB                     <1>       add    ebx, ecx ; PDE offset (for the parent)
3723 00005BED 8B0B                     <1>       mov    ecx, [ebx]
3724 00005BEF F6C101                   <1>       test   cl, PDE_A_PRESENT ; present (valid) or not ?
3725 00005BF2 7428                     <1>       jz     short dapt_2 ; parent process does not use this page
3726 00005BF4 6681E100F0               <1>       and    cx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
3727                                    <1>       ; EDI = page table entry index (0-1023)
3728 00005BF9 89FA                     <1>       mov    edx, edi
3729 00005BFB 66C1E202                 <1>       shl    dx, 2 ; *4
3730 00005BFF 01CA                     <1>       add    edx, ecx ; PTE offset (for the parent)
3731 00005C01 8B1A                     <1>       mov    ebx, [edx]
3732 00005C03 F6C301                   <1>       test   bl, PTE_A_PRESENT ; present or not ?
3733 00005C06 7414                     <1>       jz     short dapt_2 ; parent process does not use this page
3734 00005C08 662500F0                 <1>       and    ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
3735 00005C0C 6681E300F0               <1>       and    bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
3736 00005C11 39D8                     <1>       cmp    eax, ebx     ; parent's and child's pages are same ?
3737 00005C13 7507                     <1>       jne    short dapt_2 ; not same page
3738                                    <1>                         ; deallocate the child's page
3739 00005C15 800A02                   <1>       or    byte [edx], PTE_A_WRITE ; convert to writable page (parent)
3740 00005C18 59                       <1>       pop    ecx
3741 00005C19 5B                       <1>       pop    ebx
3742 00005C1A EB0D                     <1>       jmp    short dapt_4
3743                                    <1>
3744                                    <1> ; 17/04/2021
3745                                    <1> ; ('dapt_1' is disabled as temporary)
3746                                    <1> ;
3747                                    <1> ;dapt_1:
3748                                    <1> ;     or     eax, eax     ; swapped page ?
```

```
3749                                    <1> ;       jz      short dapt_5 ; no
3750                                    <1> ;                            ; yes
3751                                    <1> ;       shr     eax, 1
3752                                    <1> ;       call    unlink_swap_block ; Deallocate swapped page block
3753                                    <1> ;                                 ; on the swap disk (or in file)
3754                                    <1> ;       jmp     short dapt_5
3755                                    <1> dapt_2:
3756 00005C1C 59                       <1>         pop     ecx
3757 00005C1D 5B                       <1>         pop     ebx
3758                                    <1> dapt_3:
3759                                    <1>         ; 12/07/2016
3760 00005C1E 66A90004                  <1>         test    ax, PTE_SHARED ; shared or direct memory access indicator
3761 00005C22 7505                     <1>         jnz     short dapt_4   ; AVL bit 1 = 1, do not deallocate this page!
3762                                    <1>         ;
3763                                    <1>         ;and    ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
3764 00005C24 E814000000               <1>         call    deallocate_page ; set the mem allocation bit of this page
3765                                    <1> dapt_4:
3766 00005C29 C746FC00000000           <1>         mov     dword [esi-4], 0 ; clear/reset PTE (child, dupl. as parent)
3767                                    <1> dapt_1:     ; 17/04/2021 (temporary)
3768                                    <1> dapt_5:
3769 00005C30 47                       <1>         inc     edi ; page table entry index
3770 00005C31 81FF00040000             <1>         cmp     edi, PAGE_SIZE / 4 ; 1024
3771 00005C37 729D                     <1>         jb      short dapt_0
3772                                    <1>         ;
3773 00005C39 58                       <1>         pop     eax ; *
3774 00005C3A 5A                       <1>         pop     edx
3775 00005C3B 5F                       <1>         pop     edi
3776 00005C3C 5E                       <1>         pop     esi
3777                                    <1>         ;
3778                                    <1>         ;call   deallocate_page     ; deallocate the page table's itself
3779                                    <1>         ;retn
3780                                    <1>
3781                                    <1> deallocate_page:
3782                                    <1>         ; 15/09/2015
3783                                    <1>         ; 28/04/2015
3784                                    <1>         ; 10/03/2015
3785                                    <1>         ; 17/10/2014
3786                                    <1>         ; 12/10/2014 (Retro UNIX 386 v1 - beginning)
3787                                    <1>         ;
3788                                    <1>         ; INPUT ->
3789                                    <1>         ;       EAX = PHYSICAL (real/flat) ADDRESS OF THE ALLOCATED PAGE
3790                                    <1>         ; OUTPUT ->
3791                                    <1>         ;       [free_pages] is increased
3792                                    <1>         ;       (corresponding MEMORY ALLOCATION TABLE bit is SET)
3793                                    <1>         ;       CF = 1 if the page is already deallocated
3794                                    <1>         ;              (or not allocated) before.
3795                                    <1>         ;
3796                                    <1>         ; Modified Registers -> EAX
3797                                    <1>         ;
3798 00005C3D 53                       <1>         push    ebx
3799 00005C3E 52                       <1>         push    edx
3800                                    <1>         ;
3801 00005C3F C1E80C                   <1>         shr     eax, PAGE_SHIFT       ; shift physical address to
3802                                    <1>                                   ; 12 bits right
3803                                    <1>                                   ; to get page number
3804 00005C42 89C2                     <1>         mov     edx, eax
3805                                    <1>         ; 15/09/2015
3806 00005C44 C1EA03                   <1>         shr     edx, 3            ; to get offset to M.A.T.
3807                                    <1>                                   ; (1 allocation bit = 1 page)
3808                                    <1>                                   ; (1 allocation bytes = 8 pages)
3809 00005C47 80E2FC                   <1>         and     dl, 0FCh          ; clear lower 2 bits
3810                                    <1>                                   ; (to get 32 bit position)
3811                                    <1>         ;
3812 00005C4A BB00001000               <1>         mov     ebx, MEM_ALLOC_TBL    ; Memory Allocation Table address
3813 00005C4F 01D3                     <1>         add     ebx, edx
3814 00005C51 83E01F                   <1>         and     eax, 1Fh          ; lower 5 bits only
3815                                    <1>                                   ; (allocation bit position)
3816 00005C54 3B15[B4800100]           <1>         cmp     edx, [next_page]      ; is the new free page address lower
3817                                    <1>                                   ; than the address in 'next_page' ?
3818                                    <1>                                   ; (next/first free page value)
3819 00005C5A 7306                     <1>         jnb     short dap_1       ; no
3820 00005C5C 8915[B4800100]           <1>         mov     [next_page], edx      ; yes
3821                                    <1> dap_1:
3822 00005C62 0FAB03                   <1>         bts     [ebx], eax        ; unlink/release/deallocate page
3823                                    <1>                                   ; set relevant bit to 1.
3824                                    <1>                                   ; set CF to the previous bit value
3825                                    <1>         ;cmc                      ; complement carry flag
3826                                    <1>         ;jc     short dap_2       ; do not increase free_pages count
3827                                    <1>                                   ; if the page is already deallocated
3828                                    <1>                                   ; before.
3829 00005C65 FF05[B0800100]           <1>         inc     dword [free_pages]
3830                                    <1> dap_2:
3831 00005C6B 5A                       <1>         pop     edx
3832 00005C6C 5B                       <1>         pop     ebx
3833 00005C6D C3                       <1>         retn
3834                                    <1>
3835                                    <1> ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
3836                                    <1> ;;                                                                          ;;
3837                                    <1> ;; Copyright (C) KolibriOS team 2004-2012. All rights reserved.             ;;
3838                                    <1> ;; Distributed under terms of the GNU General Public License                ;;
3839                                    <1> ;;                                                                          ;;
3840                                    <1> ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
3841                                    <1>
3842                                    <1> ;;$Revision: 5057 $
3843                                    <1>
3844                                    <1>
3845                                    <1> ;;align 4
3846                                    <1> ;;proc alloc_page
3847                                    <1>
3848                                    <1> ;;      pushfd
3849                                    <1> ;;      cli
3850                                    <1> ;;      push    ebx
3851                                    <1> ;;;//-
3852                                    <1> ;;      cmp     [pg_data.pages_free], 1
3853                                    <1> ;;      jle     .out_of_memory
```

```
                                <1> ;;;//-
                                <1> ;;
                                <1> ;;          mov     ebx, [page_start]
                                <1> ;;          mov     ecx, [page_end]
                                <1> ;;.l1:
                                <1> ;;          bsf     eax, [ebx];
                                <1> ;;          jnz     .found
                                <1> ;;          add     ebx, 4
                                <1> ;;          cmp     ebx, ecx
                                <1> ;;          jb      .l1
                                <1> ;;          pop     ebx
                                <1> ;;          popfd
                                <1> ;;          xor     eax, eax
                                <1> ;;          ret
                                <1> ;;.found:
                                <1> ;;;//-
                                <1> ;;          dec     [pg_data.pages_free]
                                <1> ;;          jz      .out_of_memory
                                <1> ;;;//-
                                <1> ;;          btr     [ebx], eax
                                <1> ;;          mov     [page_start], ebx
                                <1> ;;          sub     ebx, sys_pgmap
                                <1> ;;          lea     eax, [eax+ebx*8]
                                <1> ;;          shl     eax, 12
                                <1> ;;;//-          dec [pg_data.pages_free]
                                <1> ;;          pop     ebx
                                <1> ;;          popfd
                                <1> ;;          ret
                                <1> ;;;//-
                                <1> ;;.out_of_memory:
                                <1> ;;          mov     [pg_data.pages_free], 1
                                <1> ;;          xor     eax, eax
                                <1> ;;          pop     ebx
                                <1> ;;          popfd
                                <1> ;;          ret
                                <1> ;;;//-
                                <1> ;;endp
                                <1>
                                <1> duplicate_page_dir:
                                <1>         ; 21/09/2015
                                <1>         ; 31/08/2015
                                <1>         ; 20/07/2015
                                <1>         ; 28/04/2015
                                <1>         ; 27/04/2015
                                <1>         ; 18/04/2015
                                <1>         ; 12/04/2015
                                <1>         ; 18/10/2014
                                <1>         ; 16/10/2014 (Retro UNIX 386 v1 - beginning)
                                <1>         ;
                                <1>         ; INPUT ->
                                <1>         ;       [u.pgdir] = PHYSICAL (real/flat) ADDRESS of the parent's
                                <1>         ;                   page directory.
                                <1>         ; OUTPUT ->
                                <1>         ;       EAX =  PHYSICAL (real/flat) ADDRESS of the child's
                                <1>         ;              page directory.
                                <1>         ;       (New page directory with new page table entries.)
                                <1>         ;       (New page tables with read only copies of the parent's
                                <1>         ;       pages.)
                                <1>         ;       EAX = 0 -> Error (CF = 1)
                                <1>         ;
                                <1>         ; Modified Registers -> none (except EAX)
                                <1>         ;
00005C6E E802FEFFFF             <1>         call    allocate_page
00005C73 723E                   <1>         jc      short dpd_err
                                <1>         ;
00005C75 55                     <1>         push    ebp ; 20/07/2015
00005C76 56                     <1>         push    esi
00005C77 57                     <1>         push    edi
00005C78 53                     <1>         push    ebx
00005C79 51                     <1>         push    ecx
00005C7A 8B35[B8030300]         <1>         mov     esi, [u.pgdir]
00005C80 89C7                   <1>         mov     edi, eax
00005C82 50                     <1>         push    eax ; save child's page directory address
                                <1>         ; 31/08/2015
                                <1>         ; copy PDE 0 from the parent's page dir to the child's page dir
                                <1>         ; (use same system space for all user page tables)
00005C83 A5                     <1>         movsd
00005C84 BD00004000             <1>         mov     ebp, 1024*4096 ; pass the 1st 4MB (system space)
00005C89 B9FF030000             <1>         mov     ecx, (PAGE_SIZE / 4) - 1 ; 1023
                                <1> dpd_0:
00005C8E AD                     <1>         lodsd
                                <1>         ;or     eax, eax
                                <1>         ;jnz    short dpd_1
00005C8F A801                   <1>         test    al, PDE_A_PRESENT ;  bit 0 =  1
00005C91 7508                   <1>         jnz     short dpd_1
                                <1>         ; 20/07/2015 (virtual address at the end of the page table)
00005C93 81C500004000           <1>         add     ebp, 1024*4096 ; page size * PTE count
00005C99 EB0F                   <1>         jmp     short dpd_2
                                <1> dpd_1:
00005C9B 662500F0               <1>         and     ax, PDE_A_CLEAR ; 0F000h ; clear attribute bits
00005C9F 89C3                   <1>         mov     ebx, eax
                                <1>         ; EBX = Parent's page table address
00005CA1 E81F000000             <1>         call    duplicate_page_table
00005CA6 720C                   <1>         jc      short dpd_p_err
                                <1>         ; EAX = Child's page table address
00005CA8 0C07                   <1>         or      al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER
                                <1>                         ; set bit 0, bit 1 and bit 2 to 1
                                <1>                         ; (present, writable, user)
                                <1> dpd_2:
00005CAA AB                     <1>         stosd
00005CAB E2E1                   <1>         loop    dpd_0
                                <1>         ;
00005CAD 58                     <1>         pop     eax  ; restore child's page directory address
                                <1> dpd_3:
00005CAE 59                     <1>         pop     ecx
```

```
3959 00005CAF 5B                  <1>       pop    ebx
3960 00005CB0 5F                  <1>       pop    edi
3961 00005CB1 5E                  <1>       pop    esi
3962 00005CB2 5D                  <1>       pop    ebp ; 20/07/2015
3963                              <1> dpd_err:
3964 00005CB3 C3                  <1>       retn
3965                              <1> dpd_p_err:
3966                              <1>       ; release the allocated pages missing (recover free space)
3967 00005CB4 58                  <1>       pop    eax  ; the new page directory address (physical)
3968 00005CB5 8B1D[B8030300]      <1>       mov    ebx, [u.pgdir] ; parent's page directory address
3969 00005CBB E8E5FEFFFF          <1>       call   deallocate_page_dir
3970 00005CC0 29C0                <1>       sub    eax, eax ; 0
3971 00005CC2 F9                  <1>       stc
3972 00005CC3 EBE9                <1>       jmp    short dpd_3
3973                              <1>
3974                              <1> duplicate_page_table:
3975                              <1>       ; 17/04/2021 - TRDOS 386 v2.0.4
3976                              <1>       ;       (temporary modifications)
3977                              <1>       ; 20/02/2017
3978                              <1>       ; 21/09/2015
3979                              <1>       ; 20/07/2015
3980                              <1>       ; 05/05/2015
3981                              <1>       ; 28/04/2015
3982                              <1>       ; 27/04/2015
3983                              <1>       ; 18/04/2015
3984                              <1>       ; 18/10/2014
3985                              <1>       ; 16/10/2014 (Retro UNIX 386 v1 - beginning)
3986                              <1>       ;
3987                              <1>       ; INPUT ->
3988                              <1>       ;      EBX = PHYSICAL (real/flat) ADDRESS of the parent's page table.
3989                              <1>       ;       20/02/2017
3990                              <1>       ;      EBP = Linear address of the page (from 'duplicate_page_dir')
3991                              <1>       ;            (Linear address = CORE + user's virtual address)
3992                              <1>       ; OUTPUT ->
3993                              <1>       ;      EAX = PHYSICAL (real/flat) ADDRESS of the child's page table.
3994                              <1>       ;            (with 'read only' attribute of page table entries)
3995                              <1>       ;      20/02/2017
3996                              <1>       ;      EBP = Next linear page address (for 'duplicate_page_dir')
3997                              <1>       ;
3998                              <1>       ;      CF = 1 -> error
3999                              <1>       ;
4000                              <1>       ; Modified Registers -> EBP (except EAX)
4001                              <1>       ;
4002 00005CC5 E8ABFDFFFF          <1>       call   allocate_page
4003 00005CCA 7258                <1>       jc     short dpt_err
4004                              <1>       ;
4005 00005CCC 50                  <1>       push   eax ; *
4006 00005CCD 56                  <1>       push   esi
4007 00005CCE 57                  <1>       push   edi
4008 00005CCF 52                  <1>       push   edx
4009 00005CD0 51                  <1>       push   ecx
4010                              <1>       ;
4011 00005CD1 89DE                <1>       mov    esi, ebx
4012 00005CD3 89C7                <1>       mov    edi, eax
4013 00005CD5 89C2                <1>       mov    edx, eax
4014 00005CD7 81C200100000        <1>       add    edx, PAGE_SIZE
4015                              <1> dpt_0:
4016 00005CDD AD                  <1>       lodsd
4017 00005CDE 21C0                <1>       and    eax, eax
4018 00005CE0 7432                <1>       jz     short dpt_3
4019 00005CE2 A801                <1>       test   al, PTE_A_PRESENT ;  bit 0 =  1
4020                              <1>       ; 17/04/2021 (temporary)
4021                              <1>       ;jnz   short dpt_1
4022 00005CE4 7439                <1>       jz     short dpt_p_err
4023                              <1>
4024                              <1> ; 17/04/2021
4025                              <1> ; ('reload_page' procedure call is disabled as temporary)
4026                              <1> ;
4027                              <1> ;     ; 20/07/2015
4028                              <1> ;     ; ebp = virtual (linear) address of the memory page
4029                              <1> ;     call   reload_page ; 28/04/2015
4030                              <1> ;     jc     short dpt_p_err
4031                              <1> dpt_1:
4032                              <1>       ; 21/09/2015
4033 00005CE6 89C1                <1>       mov    ecx, eax
4034 00005CE8 662500F0            <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
4035 00005CEC F6C102              <1>       test   cl, PTE_A_WRITE ; writable page ?
4036 00005CEF 751A                <1>       jnz    short dpt_2
4037                              <1>       ; Read only (parent) page
4038                              <1>       ;      - there is a third process which uses this page -
4039                              <1>       ; Allocate a new page for the child process
4040 00005CF1 E87FFDFFFF          <1>       call   allocate_page
4041 00005CF6 7227                <1>       jc     short dpt_p_err
4042 00005CF8 57                  <1>       push   edi
4043 00005CF9 56                  <1>       push   esi
4044 00005CFA 89CE                <1>       mov    esi, ecx
4045 00005CFC 89C7                <1>       mov    edi, eax
4046 00005CFE B900040000          <1>       mov    ecx, PAGE_SIZE/4
4047 00005D03 F3A5                <1>       rep    movsd  ; copy page (4096 bytes)
4048 00005D05 5E                  <1>       pop    esi
4049 00005D06 5F                  <1>       pop    edi
4050                              <1>       ;
4051                              <1>
4052                              <1> ; 17/04/2021
4053                              <1> ; ('add_to_swap_queue' procedure call is disabled as temporary)
4054                              <1> ;
4055                              <1> ;     push   ebx
4056                              <1> ;     push   eax
4057                              <1> ;     ; 20/07/2015
4058                              <1> ;     mov    ebx, ebp
4059                              <1> ;     ; ebx = virtual (linear) address of the memory page
4060                              <1> ;     call   add_to_swap_queue
4061                              <1> ;     pop    eax
4062                              <1> ;     pop    ebx
4063                              <1>
```

```
4064                                    <1>       ; 21/09/2015
4065 00005D07 0C07                      <1>       or     al, PTE_A_USER+PTE_A_WRITE+PTE_A_PRESENT
4066                                    <1>              ; user + writable + present page
4067 00005D09 EB09                      <1>       jmp    short dpt_3
4068                                    <1> dpt_2:
4069                                    <1>       ;or    ax, PTE_A_USER+PTE_A_PRESENT
4070 00005D0B 0C05                      <1>       or     al, PTE_A_USER+PTE_A_PRESENT
4071                                    <1>                  ; (read only page!)
4072 00005D0D 8946FC                    <1>       mov    [esi-4], eax ; update parent's PTE
4073 00005D10 660D0002                  <1>       or      ax, PTE_DUPLICATED  ; (read only page & duplicated PTE!)
4074                                    <1> dpt_3:
4075 00005D14 AB                        <1>       stosd  ; EDI points to child's PTE
4076                                    <1>       ;
4077 00005D15 81C500100000              <1>       add    ebp, 4096 ; 20/07/2015 (next page)
4078                                    <1>       ;
4079 00005D1B 39D7                      <1>       cmp    edi, edx
4080 00005D1D 72BE                      <1>       jb     short dpt_0
4081                                    <1> dpt_p_err:
4082 00005D1F 59                        <1>       pop    ecx
4083 00005D20 5A                        <1>       pop    edx
4084 00005D21 5F                        <1>       pop    edi
4085 00005D22 5E                        <1>       pop    esi
4086 00005D23 58                        <1>       pop    eax ; *
4087                                    <1> dpt_err:
4088 00005D24 C3                        <1>       retn
4089                                    <1>
4090                                    <1> page_fault_handler:       ; CPU EXCEPTION 0Eh (14) : Page Fault !
4091                                    <1>       ; 17/04/2021 - TRDOS 386 v2.0.4
4092                                    <1>       ;       (temporary modifications)
4093                                    <1>       ; 21/09/2015
4094                                    <1>       ; 19/09/2015
4095                                    <1>       ; 17/09/2015
4096                                    <1>       ; 28/08/2015
4097                                    <1>       ; 20/07/2015
4098                                    <1>       ; 28/06/2015
4099                                    <1>       ; 03/05/2015
4100                                    <1>       ; 30/04/2015
4101                                    <1>       ; 18/04/2015
4102                                    <1>       ; 12/04/2015
4103                                    <1>       ; 30/10/2014
4104                                    <1>       ; 11/09/2014
4105                                    <1>       ; 10/09/2014 (Retro UNIX 386 v1 - beginning)
4106                                    <1>       ;
4107                                    <1>       ; Note: This is not an interrupt/exception handler.
4108                                    <1>       ;     This is a 'page fault remedy' subroutine
4109                                    <1>       ;     which will be called by standard/uniform
4110                                    <1>       ;     exception handler.
4111                                    <1>       ;
4112                                    <1>       ; INPUT ->
4113                                    <1>       ;     [error_code] = 32 bit ERROR CODE (lower 5 bits are valid)
4114                                    <1>       ;
4115                                    <1>       ;     cr2 = the virtual (linear) address
4116                                    <1>       ;           which has caused to page fault (19/09/2015)
4117                                    <1>       ;
4118                                    <1>       ; OUTPUT ->
4119                                    <1>       ;     (corresponding PAGE TABLE ENTRY is mapped/set)
4120                                    <1>       ;     EAX = 0 -> no error
4121                                    <1>       ;     EAX > 0 -> error code in EAX (also CF = 1)
4122                                    <1>       ;
4123                                    <1>       ; Modified Registers -> none (except EAX)
4124                                    <1>       ;
4125                                    <1>       ;
4126                                    <1>       ; ERROR CODE:
4127                                    <1>       ;      31  .....   4   3  2   1  0
4128                                    <1>       ;     +---+-- --+---+---+---+---+---+
4129                                    <1>       ;     |   Reserved  | I | R | U | W | P |
4130                                    <1>       ;     +---+-- --+---+---+---+---+---+
4131                                    <1>       ;
4132                                    <1>       ; P : PRESENT -     When set, the page fault was caused by
4133                                    <1>       ;             a page-protection violation. When not set,
4134                                    <1>       ;             it was caused by a non-present page.
4135                                    <1>       ; W : WRITE   -     When set, the page fault was caused by
4136                                    <1>       ;             a page write. When not set, it was caused
4137                                    <1>       ;             by a page read.
4138                                    <1>       ; U : USER    -     When set, the page fault was caused
4139                                    <1>       ;             while CPL = 3.
4140                                    <1>       ;             This does not necessarily mean that
4141                                    <1>       ;             the page fault was a privilege violation.
4142                                    <1>       ; R : RESERVD -     When set, the page fault was caused by
4143                                    <1>       ;     WRITE  reading a 1 in a reserved field.
4144                                    <1>       ; I : INSTRUC -     When set, the page fault was caused by
4145                                    <1>       ;     FETCH  an instruction fetch
4146                                    <1>       ;
4147                                    <1>       ;; x86 (32 bit) VIRTUAL ADDRESS TRANSLATION
4148                                    <1>       ;  31               22                  12 11                      0
4149                                    <1>       ; +------------------+------------------+-------------------------+
4150                                    <1>       ; | PAGE DIR. ENTRY # | PAGE TAB. ENTRY # |        OFFSET          |
4151                                    <1>       ; +------------------+------------------+-------------------------+
4152                                    <1>       ;
4153                                    <1>
4154                                    <1>       ;; CR3 REGISTER (Control Register 3)
4155                                    <1>       ;  31                                    12             5 4 3 2   0
4156                                    <1>       ; +----------------------------------------+-------------+---+-----+
4157                                    <1>       ; |                                        |             |P|P|     |
4158                                    <1>       ; |   PAGE DIRECTORY TABLE BASE ADDRESS    |  reserved   |C|W|rsvrd|
4159                                    <1>       ; |                                        |             |D|T|     |
4160                                    <1>       ; +----------------------------------------+-------------+---+-----+
4161                                    <1>       ;
4162                                    <1>       ;     PWT    - WRITE THROUGH
4163                                    <1>       ;     PCD    - CACHE DISABLE
4164                                    <1>       ;
4165                                    <1>       ;
4166                                    <1>       ;; x86 PAGE DIRECTORY ENTRY (4 KByte Page)
4167                                    <1>       ;  31                                    12 11  9 8 7 6 5 4 3 2 1 0
4168                                    <1>       ; +----------------------------------------+-----+---+-+-+---+-+-+-+
```

```
                        <1>       ; |                                       |     | | | | |P|P|U|R| |
                        <1>       ; |     PAGE TABLE BASE ADDRESS 31..12    | AVL |G|0|D|A|C|W|/|/|P|
                        <1>       ; |                                       |     | | | | |D|T|S|W| |
                        <1>       ; +---------------------------------------+-----+---+-+-+---+-+-+-+
                        <1>       ;
                        <1>       ;     P   - PRESENT
                        <1>       ;     R/W - READ/WRITE
                        <1>       ;     U/S - USER/SUPERVISOR
                        <1>       ;      PWT - WRITE THROUGH
                        <1>       ;      PCD - CACHE DISABLE
                        <1>       ;      A   - ACCESSED
                        <1>       ;     D   - DIRTY (IGNORED)
                        <1>       ;      PAT - PAGE ATTRIBUTE TABLE INDEX (CACHE BEHAVIOR)
                        <1>       ;      G   - GLOBAL (IGNORED)
                        <1>       ;     AVL - AVAILABLE FOR SYSTEMS PROGRAMMER USE
                        <1>       ;
                        <1>       ;
                        <1>       ;; x86 PAGE TABLE ENTRY (4 KByte Page)
                        <1>       ;  31                                  12 11  9 8 7 6 5 4 3 2 1 0
                        <1>       ; +---------------------------------------+-----+---+-+-+---+-+-+-+
                        <1>       ; |                                       |     | |P| | |P|P|U|R| |
                        <1>       ; |     PAGE FRAME BASE ADDRESS 31..12    | AVL |G|A|D|A|C|W|/|/|P|
                        <1>       ; |                                       |     | |T| | |D|T|S|W| |
                        <1>       ; +---------------------------------------+-----+---+-+-+---+-+-+-+
                        <1>       ;
                        <1>       ;     P   - PRESENT
                        <1>       ;     R/W - READ/WRITE
                        <1>       ;     U/S - USER/SUPERVISOR
                        <1>       ;      PWT - WRITE THROUGH
                        <1>       ;      PCD - CACHE DISABLE
                        <1>       ;      A   - ACCESSED
                        <1>       ;     D   - DIRTY
                        <1>       ;      PAT - PAGE ATTRIBUTE TABLE INDEX (CACHE BEHAVIOR)
                        <1>       ;      G   - GLOBAL
                        <1>       ;     AVL - AVAILABLE FOR SYSTEMS PROGRAMMER USE
                        <1>       ;
                        <1>       ;
                        <1>       ;; 80386 PAGE TABLE ENTRY (4 KByte Page)
                        <1>       ;  31                                  12 11  9 8 7 6 5 4 3 2 1 0
                        <1>       ; +---------------------------------------+-----+-+-+-+-+---+-+-+-+
                        <1>       ; |                                       |     | | | | | | |U|R| |
                        <1>       ; |     PAGE FRAME BASE ADDRESS 31..12    | AVL |0|0|D|A|0|0|/|/|P|
                        <1>       ; |                                       |     | | | | | | |S|W| |
                        <1>       ; +---------------------------------------+-----+-+-+-+-+---+-+-+-+
                        <1>       ;
                        <1>       ;       P   - PRESENT
                        <1>       ;       R/W - READ/WRITE
                        <1>       ;       U/S - USER/SUPERVISOR
                        <1>       ;       D   - DIRTY
                        <1>       ;       AVL - AVAILABLE FOR SYSTEMS PROGRAMMER USE
                        <1>       ;
                        <1>       ;       NOTE: 0 INDICATES INTEL RESERVED. DO NOT DEFINE.
                        <1>       ;
                        <1>       ;
                        <1>       ;; Invalid Page Table Entry
                        <1>       ; 31                                                          1 0
                        <1>       ; +-------------------------------------------------------------+-+
                        <1>       ; |                                                             | |
                        <1>       ; |                           AVAILABLE                         |0|
                        <1>       ; |                                                             | |
                        <1>       ; +-------------------------------------------------------------+-+
                        <1>       ;
                        <1>
00005D25 53             <1>       push  ebx
00005D26 52             <1>       push  edx
00005D27 51             <1>       push  ecx
                        <1>       ;
                        <1>       ; 21/09/2015 (debugging)
00005D28 FF05[CC030300] <1>       inc   dword [u.pfcount] ; page fault count for running process
00005D2E FF05[68050300] <1>       inc   dword [PF_Count] ; total page fault count
                        <1>       ; 28/06/2015
                        <1>       ;mov   edx, [error_code] ; Lower 5 bits are valid
00005D34 8A15[60050300] <1>       mov   dl, [error_code]
                        <1>       ;
00005D3A F6C201         <1>       test  dl, 1  ; page fault was caused by a non-present page
                        <1>                    ; sign
00005D3D 741A           <1>       jz    short pfh_alloc_np
                        <1>       ;
                        <1>       ; If it is not a 'write on read only page' type page fault
                        <1>       ; major page fault error with minor reason must be returned without
                        <1>       ; fixing the problem. 'sys_exit with error' will be needed
                        <1>       ; after return here!
                        <1>       ; Page fault will be remedied, by copying page contents
                        <1>       ; to newly allocated page with write permission;
                        <1>       ; sys_fork -> sys_exec -> copy on write, demand paging method is
                        <1>       ; used for working with minimum possible memory usage.
                        <1>       ; sys_fork will duplicate page directory and tables of parent
                        <1>       ; process with 'read only' flag. If the child process attempts to
                        <1>       ; write on these read only pages, page fault will be directed here
                        <1>       ; for allocating a new page with same data/content.
                        <1>       ;
                        <1>       ; IMPORTANT : Retro UNIX 386 v1 (and SINGLIX and TR-DOS)
                        <1>       ; will not force to separate CODE and DATA space
                        <1>       ; in a process/program...
                        <1>       ; CODE segment/section may contain DATA!
                        <1>       ; It is flat, smoth and simplest programming method already as in
                        <1>       ; Retro UNIX 8086 v1 and MS-DOS programs.
                        <1>       ;
00005D3F F6C202         <1>       test  dl, 2  ; page fault was caused by a page write
                        <1>                    ; sign
00005D42 0F8481000000   <1>       jz    pfh_p_err
                        <1>       ; 31/08/2015
00005D48 F6C204         <1>       test  dl, 4  ; page fault was caused while CPL = 3 (user mode)
                        <1>                    ; sign.  (U+W+P = 4+2+1 = 7)
00005D4B 747C           <1>       jz    pfh_pv_err
```

```
                        <1>       ;
                        <1>       ; make a new page and copy the parent's page content
                        <1>       ; as the child's new page content
                        <1>       ;
00005D4D 0F20D3         <1>       mov    ebx, cr2 ; CR2 contains the linear address
                        <1>                      ; which has caused to page fault
00005D50 E87C000000     <1>       call   copy_page
00005D55 726B           <1>       jc    pfh_im_err ; insufficient memory
                        <1>       ;
00005D57 EB63           <1>       jmp   pfh_cpp_ok
                        <1>       ;
                        <1> pfh_alloc_np:
00005D59 E817FDFFFF     <1>       call   allocate_page; (allocate a new page)
00005D5E 7262           <1>         jc      pfh_im_err      ; 'insufficient memory' error
                        <1> pfh_chk_cpl:
                        <1>       ; EAX = Physical (base) address of the allocated (new) page
                        <1>             ; (Lower 12 bits are ZERO, because
                        <1>             ;      the address is on a page boundary)
00005D60 80E204         <1>       and    dl, 4  ; CPL = 3 ?
00005D63 7505           <1>       jnz    short pfh_um
                        <1>                     ; Page fault handler for kernel/system mode (CPL=0)
00005D65 0F20DB         <1>       mov    ebx, cr3 ; CR3 (Control Register 3) contains physical address
                        <1>                      ; of the current/active page directory
                        <1>                      ; (Always kernel/system mode page directory, here!)
                        <1>                      ; Note: Lower 12 bits are 0. (page boundary)
00005D68 EB06           <1>       jmp    short pfh_get_pde
                        <1>       ;
                        <1> pfh_um:                     ; Page fault handler for user/appl. mode (CPL=3)
00005D6A 8B1D[B8030300] <1>       mov    ebx, [u.pgdir] ; Page directory of current/active process
                        <1>                     ; Physical address of the USER's page directory
                        <1>                     ; Note: Lower 12 bits are 0. (page boundary)
                        <1> pfh_get_pde:
00005D70 80CA03         <1>       or     dl, 3  ; USER + WRITE + PRESENT or SYSTEM + WRITE + PRESENT
00005D73 0F20D1         <1>       mov    ecx, cr2 ; CR2 contains the virtual address
                        <1>                      ; which has been caused to page fault
                        <1>                      ;
00005D76 C1E914         <1>       shr    ecx, 20       ; shift 20 bits right
00005D79 80E1FC         <1>       and    cl, 0FCh ; mask lower 2 bits to get PDE offset
                        <1>       ;
00005D7C 01CB           <1>       add    ebx, ecx ; now, EBX points to the relevant page dir entry
00005D7E 8B0B           <1>       mov    ecx, [ebx] ; physical (base) address of the page table
00005D80 F6C101         <1>       test   cl, 1   ; check bit 0 is set (1) or not (0).
00005D83 740B           <1>       jz     short pfh_set_pde ; Page directory entry is not valid,
                        <1>                              ; set/validate page directory entry
00005D85 6681E100F0     <1>       and    cx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
00005D8A 89CB           <1>       mov    ebx, ecx ; Physical address of the page table
00005D8C 89C1           <1>       mov    ecx, eax ; new page address (physical)
00005D8E EB16           <1>       jmp    short pfh_get_pte
                        <1> pfh_set_pde:
                        <1>       ;; NOTE: Page directories and page tables never be swapped out!
                        <1>       ;;      (So, we know this PDE is empty or invalid)
                        <1>       ;
00005D90 08D0           <1>       or     al, dl  ; lower 3 bits are used as U/S, R/W, P flags
00005D92 8903           <1>       mov    [ebx], eax ; Let's put the new page directory entry here !
00005D94 30C0           <1>       xor    al, al  ; clear lower (3..8) bits
00005D96 89C3           <1>       mov    ebx, eax
00005D98 E8D8FCFFFF     <1>       call   allocate_page ; (allocate a new page)
00005D9D 7223           <1>       jc     short pfh_im_err   ; 'insufficient memory' error
                        <1> pfh_spde_1:
                        <1>       ; EAX = Physical (base) address of the allocated (new) page
00005D9F 89C1           <1>       mov    ecx, eax
00005DA1 E840FDFFFF     <1>       call   clear_page ; Clear page content
                        <1> pfh_get_pte:
00005DA6 0F20D0         <1>       mov    eax, cr2 ; virtual address
                        <1>                      ; which has been caused to page fault
00005DA9 89C7           <1>       mov    edi, eax ; 20/07/2015
00005DAB C1E80C         <1>       shr    eax, 12       ; shift 12 bit right to get
                        <1>                      ; higher 20 bits of the page fault address
00005DAE 25FF030000     <1>       and    eax, 3FFh ; mask PDE# bits, the result is PTE# (0 to 1023)
00005DB3 C1E002         <1>       shl    eax, 2 ; shift 2 bits left to get PTE offset
00005DB6 01C3           <1>       add    ebx, eax ; now, EBX points to the relevant page table entry
                        <1> ; 17/04/2021 temporary
                        <1> ;     mov    eax, [ebx] ; get previous value of pte
                        <1> ;            ; bit 0 of EAX is always 0 (otherwise we would not be here)
                        <1>
                        <1> ; 17/04/2021
                        <1> ; ('swap_in' procedure call has been disabled as temporary)
                        <1> ;
                        <1> ;     and    eax, eax
                        <1> ;     jz     short pfh_gpte_1
                        <1> ;     ; 20/07/2015
                        <1> ;     xchg   ebx, ecx ; new page address (physical)
                        <1> ;     push   ebp ; 20/07/2015
                        <1> ;     mov    ebp, cr2
                        <1> ;            ; ECX = physical address of the page table entry
                        <1> ;            ; EBX = Memory page address (physical!)
                        <1> ;            ; EAX = Swap disk (offset) address
                        <1> ;            ; EBP = virtual address (page fault address)
                        <1> ;     call   swap_in
                        <1> ;     pop    ebp
                        <1> ;     jc      short pfh_err_retn
                        <1> ;     xchg   ecx, ebx
                        <1> ;            ; EBX = physical address of the page table entry
                        <1> ;            ; ECX = new page
                        <1> pfh_gpte_1:
00005DB8 08D1           <1>       or     cl, dl ; lower 3 bits are used as U/S, R/W, P flags
00005DBA 890B           <1>       mov    [ebx], ecx ; Let's put the new page table entry here !
                        <1> pfh_cpp_ok:
                        <1> ; 17/04/2021
                        <1> ; ('add_to_swap_queue' procedure call has been disabled as temporary)
                        <1> ;
                        <1> ;     ; 20/07/2015
                        <1> ;     mov    ebx, cr2
                        <1> ;     call   add_to_swap_queue
                        <1>       ;
```

```
4379                                    <1>       ; The new PTE (which contains the new page) will be added to
4380                                    <1>       ; the swap queue, here.
4381                                    <1>       ; (Later, if memory will become insufficient,
4382                                    <1>       ; one page will be swapped out which is at the head of
4383                                    <1>       ; the swap queue by using FIFO and access check methods.)
4384                                    <1>       ;
4385 00005DBC 31C0                      <1>       xor    eax, eax  ; 0
4386                                    <1>       ;
4387                                    <1> pfh_err_retn:
4388 00005DBE 59                       <1>       pop    ecx
4389 00005DBF 5A                       <1>       pop    edx
4390 00005DC0 5B                       <1>       pop    ebx
4391 00005DC1 C3                       <1>       retn
4392                                    <1>
4393                                    <1> pfh_im_err:
4394 00005DC2 B8E4000000               <1>       mov    eax, ERR_MAJOR_PF + ERR_MINOR_IM ; Error code in AX
4395                                    <1>                     ; Major (Primary) Error: Page Fault
4396                                    <1>                     ; Minor (Secondary) Error: Insufficient Memory !
4397 00005DC7 EBF5                     <1>       jmp    short pfh_err_retn
4398                                    <1>
4399                                    <1> pfh_p_err: ; 09/03/2015
4400                                    <1> pfh_pv_err:
4401                                    <1>       ; Page fault was caused by a protection-violation
4402 00005DC9 B8E6000000               <1>       mov    eax, ERR_MAJOR_PF + ERR_MINOR_PV ; Error code in AX
4403                                    <1>                     ; Major (Primary) Error: Page Fault
4404                                    <1>                     ; Minor (Secondary) Error: Protection violation !
4405 00005DCE F9                       <1>       stc
4406 00005DCF EBED                     <1>       jmp    short pfh_err_retn
4407                                    <1>
4408                                    <1> copy_page:
4409                                    <1>       ; 22/09/2015
4410                                    <1>       ; 21/09/2015
4411                                    <1>       ; 19/09/2015
4412                                    <1>       ; 07/09/2015
4413                                    <1>       ; 31/08/2015
4414                                    <1>       ; 20/07/2015
4415                                    <1>       ; 05/05/2015
4416                                    <1>       ; 03/05/2015
4417                                    <1>       ; 18/04/2015
4418                                    <1>       ; 12/04/2015
4419                                    <1>       ; 30/10/2014
4420                                    <1>       ; 18/10/2014 (Retro UNIX 386 v1 - beginning)
4421                                    <1>       ;
4422                                    <1>       ; INPUT ->
4423                                    <1>       ;      EBX = Virtual (linear) address of source page
4424                                    <1>       ;           (Page fault address)
4425                                    <1>       ; OUTPUT ->
4426                                    <1>       ;      EAX = PHYSICAL (real/flat) ADDRESS OF THE ALLOCATED PAGE
4427                                    <1>       ;      (corresponding PAGE TABLE ENTRY is mapped/set)
4428                                    <1>       ;      EAX = 0 (CF = 1)
4429                                    <1>       ;            if there is not a free page to be allocated
4430                                    <1>       ;      (page content of the source page will be copied
4431                                    <1>       ;      onto the target/new page)
4432                                    <1>       ;
4433                                    <1>       ; Modified Registers -> ecx, ebx (except EAX)
4434                                    <1>       ;
4435 00005DD1 56                       <1>       push   esi
4436 00005DD2 57                       <1>       push   edi
4437                                    <1>       ;push  ebx
4438                                    <1>       ;push  ecx
4439 00005DD3 31F6                     <1>       xor    esi, esi
4440 00005DD5 C1EB0C                   <1>       shr    ebx, 12 ; shift 12 bits right to get PDE & PTE numbers
4441 00005DD8 89D9                     <1>       mov    ecx, ebx ; save page fault address (as 12 bit shifted)
4442 00005DDA C1EB08                   <1>       shr    ebx, 8  ; shift 8 bits right and then
4443 00005DDD 80E3FC                   <1>       and    bl, 0FCh ; mask lower 2 bits to get PDE offset
4444 00005DE0 89DF                     <1>       mov    edi, ebx ; save it for the parent of current process
4445 00005DE2 031D[B8030300]           <1>       add    ebx, [u.pgdir] ; EBX points to the relevant page dir entry
4446 00005DE8 8B03                     <1>       mov    eax, [ebx] ; physical (base) address of the page table
4447 00005DEA 662500F0                 <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
4448 00005DEE 89CB                     <1>       mov    ebx, ecx   ; (restore higher 20 bits of page fault address)
4449 00005DF0 81E3FF030000             <1>       and    ebx, 3FFh  ; mask PDE# bits, the result is PTE# (0 to 1023)
4450 00005DF6 66C1E302                 <1>       shl    bx, 2     ; shift 2 bits left to get PTE offset
4451 00005DFA 01C3                     <1>       add    ebx, eax   ; EBX points to the relevant page table entry
4452                                    <1>       ; 07/09/2015
4453 00005DFC 66F7030002               <1>         test    word [ebx], PTE_DUPLICATED ; (Does current process share this
4454                                    <1>                                ; read only page as a child process?)
4455 00005E01 7509                     <1>       jnz    short cpp_0 ; yes
4456 00005E03 8B0B                     <1>       mov    ecx, [ebx] ; PTE value
4457 00005E05 6681E100F0               <1>       and    cx, PTE_A_CLEAR ; 0F000h  ; clear page attributes
4458 00005E0A EB32                     <1>       jmp    short cpp_1
4459                                    <1> cpp_0:
4460 00005E0C 89FE                     <1>       mov    esi, edi
4461 00005E0E 0335[BC030300]           <1>       add    esi, [u.ppgdir] ; the parent's page directory entry
4462 00005E14 8B06                     <1>       mov    eax, [esi] ; physical (base) address of the page table
4463 00005E16 662500F0                 <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
4464 00005E1A 89CE                     <1>       mov    esi, ecx   ; (restore higher 20 bits of page fault address)
4465 00005E1C 81E6FF030000             <1>       and    esi, 3FFh  ; mask PDE# bits, the result is PTE# (0 to 1023)
4466 00005E22 66C1E602                 <1>       shl    si, 2     ; shift 2 bits left to get PTE offset
4467 00005E26 01C6                     <1>       add    esi, eax   ; EDX points to the relevant page table entry
4468 00005E28 8B0E                     <1>       mov    ecx, [esi] ; PTE value of the parent process
4469                                    <1>       ; 21/09/2015
4470 00005E2A 8B03                     <1>       mov    eax, [ebx] ; PTE value of the child process
4471 00005E2C 662500F0                 <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear page attributes
4472                                    <1>       ;
4473 00005E30 F6C101                   <1>       test   cl, PTE_A_PRESENT ; is it a present/valid page ?
4474 00005E33 7424                     <1>       jz     short cpp_3 ; the parent's page is not same page
4475                                    <1>       ;
4476 00005E35 6681E100F0               <1>       and    cx, PTE_A_CLEAR ; 0F000h ; clear page attributes
4477 00005E3A 39C8                     <1>       cmp    eax, ecx   ; Same page?
4478 00005E3C 751B                     <1>       jne    short cpp_3 ; Parent page and child page are not same
4479                                    <1>                        ; Convert child's page to writable page
4480                                    <1> cpp_1:
4481 00005E3E E832FCFFFF               <1>       call   allocate_page
4482 00005E43 721A                     <1>       jc     short cpp_4 ; 'insufficient memory' error
4483 00005E45 21F6                     <1>       and    esi, esi    ; check ESI is valid or not
```

```
4484 00005E47 7405                <1>       jz     short cpp_2
4485                              <1>              ; Convert read only page to writable page
4486                              <1>              ;(for the parent of the current process)
4487                              <1>       ;and   word [esi], PTE_A_CLEAR ; 0F000h
4488                              <1>       ; 22/09/2015
4489 00005E49 890E                <1>       mov    [esi], ecx
4490 00005E4B 800E07              <1>       or     byte [esi], PTE_A_PRESENT + PTE_A_WRITE + PTE_A_USER
4491                              <1>                            ; 1+2+4 = 7
4492                              <1> cpp_2:
4493 00005E4E 89C7                <1>       mov    edi, eax ; new page address of the child process
4494                              <1>       ; 07/09/2015
4495 00005E50 89CE                <1>       mov    esi, ecx ; the page address of the parent process
4496 00005E52 B900040000          <1>       mov    ecx, PAGE_SIZE / 4
4497 00005E57 F3A5                <1>       rep    movsd ; 31/08/2015
4498                              <1> cpp_3:
4499 00005E59 0C07                <1>       or     al, PTE_A_PRESENT + PTE_A_WRITE + PTE_A_USER ; 1+2+4 = 7
4500 00005E5B 8903                <1>       mov    [ebx], eax ; Update PTE
4501 00005E5D 28C0                <1>       sub    al, al ; clear attributes
4502                              <1> cpp_4:
4503                              <1>       ;pop   ecx
4504                              <1>       ;pop   ebx
4505 00005E5F 5F                  <1>       pop    edi
4506 00005E60 5E                  <1>       pop    esi
4507 00005E61 C3                  <1>       retn
4508                              <1>
4509                              <1> ;; 28/04/2015
4510                              <1> ;; 24/10/2014
4511                              <1> ;; 21/10/2014 (Retro UNIX 386 v1 - beginning)
4512                              <1> ;; SWAP_PAGE_QUEUE (4096 bytes)
4513                              <1> ;;
4514                              <1> ;;   0000   0001   0002   0003   ....   1020   1021   1022   1023
4515                              <1> ;; +------+------+------+------+-    -+------+------+------+------+
4516                              <1> ;; |  pg1 |  pg2 |  pg3 |  pg4 | .... |pg1021|pg1022|pg1023|pg1024|
4517                              <1> ;; +------+------+------+------+-    -+------+------+------+------+
4518                              <1> ;;
4519                              <1> ;; [swpq_last] = 0 to 4096 (step 4) -> the last position on the queue
4520                              <1> ;;
4521                              <1> ;; Method:
4522                              <1> ;;    Swap page queue is a list of allocated pages with physical
4523                              <1> ;;    addresses (system mode virtual adresses = physical addresses).
4524                              <1> ;;    It is used for 'swap_in' and 'swap_out' procedures.
4525                              <1> ;;    When a new page is being allocated, swap queue is updated
4526                              <1> ;;    by 'swap_queue_shift' procedure, header of the queue (offset 0)
4527                              <1> ;;    is checked for 'accessed' flag. If the 1st page on the queue
4528                              <1> ;;    is 'accessed' or 'read only', it is dropped from the list;
4529                              <1> ;;    other pages from the 2nd to the last (in [swpq_last]) shifted
4530                              <1> ;;    to head then the 2nd page becomes the 1st and '[swpq_last]'
4531                              <1> ;;    offset value becomes it's previous offset value - 4.
4532                              <1> ;;    If the 1st page of the swap page queue is not 'accessed'
4533                              <1> ;;    the queue/list is not shifted.
4534                              <1> ;;    After the queue/list shift, newly allocated page is added
4535                              <1> ;;    to the tail of the queue at the [swpq_count*4] position.
4536                              <1> ;;    But, if [swpq_count] > 1023, the newly allocated page
4537                              <1> ;;    will not be added to the tail of swap page queue.
4538                              <1> ;;
4539                              <1> ;;    During 'swap_out' procedure, swap page queue is checked for
4540                              <1> ;;    the first non-accessed, writable page in the list,
4541                              <1> ;;    from the head to the tail. The list is shifted to left
4542                              <1> ;;    (to the head) till a non-accessed page will be found in the list.
4543                              <1> ;;    Then, this page     is swapped out (to disk) and then it is dropped
4544                              <1> ;;    from the list by a final swap queue shift. [swpq_count] value
4545                              <1> ;;    is changed. If all pages on the queue' are 'accessed',
4546                              <1> ;;    'insufficient memory' error will be returned ('swap_out'
4547                              <1> ;;    procedure will be failed)...
4548                              <1> ;;
4549                              <1> ;;    Note: If the 1st page of the queue is an 'accessed' page,
4550                              <1> ;;    'accessed' flag of the page will be reset (0) and that page
4551                              <1> ;;    (PTE) will be added to the tail of the queue after
4552                              <1> ;;    the check, if [swpq_count] < 1023. If [swpq_count] = 1024
4553                              <1> ;;    the queue will be rotated and the PTE in the head will be
4554                              <1> ;;    added to the tail after resetting 'accessed' bit.
4555                              <1> ;;
4556                              <1> ;;
4557                              <1> ;;
4558                              <1> ;; SWAP DISK/FILE (with 4096 bytes swapped page blocks)
4559                              <1> ;;
4560                              <1> ;;  00000000  00000004  00000008  0000000C   ...   size-8    size-4
4561                              <1> ;; +---------+---------+---------+---------+-- --+---------+---------+
4562                              <1> ;; |descriptr| page(1) | page(2) | page(3) | ... |page(n-1)| page(n) |
4563                              <1> ;; +---------+---------+---------+---------+-- --+---------+---------+
4564                              <1> ;;
4565                              <1> ;; [swpd_next] = the first free block address in swapped page records
4566                              <1> ;;                  for next free block search by 'swap_out' procedure.
4567                              <1> ;; [swpd_size] = swap disk/file size in sectors (512 bytes)
4568                              <1> ;;           NOTE: max. possible swap disk size is 1024 GB
4569                              <1> ;;           (entire swap space must be accessed by using
4570                              <1> ;;           31 bit offset address)
4571                              <1> ;; [swpd_free] = free block (4096 bytes) count in swap disk/file space
4572                              <1> ;; [swpd_start] = absolute/start address of the swap disk/file
4573                              <1> ;;            0 for file, or beginning sector of the swap partition
4574                              <1> ;; [swp_drv] = logical drive description table addr. of swap disk/file
4575                              <1> ;;
4576                              <1> ;;
4577                              <1> ;; Method:
4578                              <1> ;;    When the memory (ram) becomes insufficient, page allocation
4579                              <1> ;;    procedure swaps out a page from memory to the swap disk
4580                              <1> ;;    (partition) or swap file to get a new free page at the memory.
4581                              <1> ;;    Swapping out is performed by using swap page queue.
4582                              <1> ;;
4583                              <1> ;;    Allocation block size of swap disk/file is equal to page size
4584                              <1> ;;    (4096 bytes). Swapping address (in sectors) is recorded
4585                              <1> ;;    into relevant page file entry as 31 bit physical (logical)
4586                              <1> ;;    offset address as 1 bit shifted to left for present flag (0).
4587                              <1> ;;    Swapped page address is between 1 and swap disk/file size - 4.
4588                              <1> ;;    Absolute physical (logical) address of the swapped page is
```

```
;;    calculated by adding offset value to the swap partition's
;;    start address. If the swap device (disk) is a virtual disk
;;    or it is a file, start address of the swap disk/volume is 0,
;;    and offset value is equal to absolute (physical or logical)
;;    address/position. (It has not to be ZERO if the swap partition
;;    is in a partitioned virtual hard disk.)
;;
;;    Note: Swap addresses are always specified/declared in sectors,
;;    not in bytes or     in blocks/zones/clusters (4096 bytes) as unit.
;;
;;    Swap disk/file allocation is mapped via 'Swap Allocation Table'
;;    at memory as similar to 'Memory Allocation Table'.
;;
;;    Every bit of Swap Allocation Table repsesents one swap block
;;    (equal to page size) respectively. Bit 0 of the S.A.T. byte 0
;;    is reserved for swap disk/file block 0 as descriptor block
;;    (also for compatibility with PTE). If bit value is ZERO,
;;    it means relevant (respective) block is in use, and,
;;    of course, if bit value is 1, it means relevant (respective)
;;      swap disk/file block is free.
;;    For example: bit 1 of the byte 128 repsesents block 1025
;;    (128*8+1) or sector (offset) 8200 on the swap disk or
;;    byte (offset/position) 4198400 in the swap file.
;;    4GB swap space is represented via 128KB Swap Allocation Table.
;;    Initial layout of Swap Allocation Table is as follows:
;;    ------------------------------------------------------------
;;    01111111111111111111111 .... 11111111111111111111111111111
;;    ------------------------------------------------------------
;;    (0 is reserved block, 1s represent free blocks respectively.)
;;    (Note: Allocation cell/unit of the table is bit, not byte)
;;
;;    ............................................................
;;    'swap_out' procedure checks 'free_swap_blocks' count at first,
;;    then it searches Swap Allocation Table if free count is not
;;    zero. From begining the [swpd_next] dword value, the first bit
;;    position with value of 1 on the table is converted to swap
;;    disk/file offset address, in sectors (not 4096 bytes block).
;;    'ldrv_write' procedure is called with ldrv (logical drive
;;    number of physical swap disk or virtual swap disk)
;;    number, sector offset (not absolute sector -LBA- number),
;;    and sector count (8, 512*8 = 4096) and buffer adress
;;    (memory page). That will be a direct disk write procedure.
;;    (for preventing late memory allocation, significant waiting).
;;    If disk write procedure returns with error or free count of
;;    swap blocks is ZERO, 'swap_out' procedure will return with
;;    'insufficient memory error' (cf=1).
;;
;;    (Note: Even if free swap disk/file blocks was not zero,
;;    any disk write error will not be fixed by 'swap_out' procedure,
;;    in other words, 'swap_out' will not check the table for other
;;    free blocks after a disk write error. It will return to
;;    the caller with error (CF=1) which means swapping is failed.
;;
;;    After writing the page on to swap disk/file address/sector,
;;    'swap_out' procesure returns with that swap (offset) sector
;;    address (cf=0).
;;
;;    ............................................................
;;
;;    'swap_in' procedure loads addressed (relevant) swap disk or
;;    file sectors at specified memory page. Then page allocation
;;    procedure updates relevant page table entry with 'present'
;;    attribute. If swap disk or file reading fails there is nothing
;;    to do, except to terminate the process which is the owner of
;;    the swapped page.
;;
;;    'swap_in' procedure sets the relevant/respective bit value
;;    in the Swap Allocation Table (as free block). 'swap_in' also
;;    updates [swpd_first] pointer if it is required.
;;
;;    ............................................................
;;
;;    Note: If [swap_enabled] value is ZERO, that means there is not
;;    a swap disk or swap file in use... 'swap_in' and 'swap_out'
;;    procedures ans 'swap page que' procedures will not be active...
;;    'Insufficient memory' error will be returned by 'swap_out'
;;    and 'general protection fault' will be returned by 'swap_in'
;;    procedure, if it is called mistakenly (a wrong value in a PTE).
;;
;
; 17/04/2021
; ('swap_in' procedure call is disabled as temporary)
;
swap_in:
      ; 31/08/2015
      ; 20/07/2015
      ; 28/04/2015
      ; 18/04/2015
      ; 24/10/2014 (Retro UNIX 386 v1 - beginning)
      ;
      ; INPUT ->
      ;     EBX = PHYSICAL (real/flat) ADDRESS OF THE MEMORY PAGE
      ;     EBP = VIRTUAL (LINEAR) ADDRESS (page fault address)
      ;     EAX = Offset Address for the swapped page on the
      ;           swap disk or in the swap file.
      ;
      ; OUTPUT ->
      ;     EAX = 0 if loading at memory has been successful
      ;
      ;     CF = 1 -> swap disk reading error (disk/file not present
      ;              or sector not present or drive not ready
      ;          EAX = Error code
      ;          [u.error] = EAX
      ;                   = The last error code for the process
```

```
4694                            <1>       ;                     (will be reset after returning to user)
4695                            <1>       ;
4696                            <1>       ; Modified Registers -> EAX
4697                            <1>       ;
4698                            <1>
4699                            <1> ;     cmp   dword [swp_drv], 0
4700                            <1> ;     jna   short swpin_dnp_err
4701                            <1> ;
4702                            <1> ;     cmp   eax, [swpd_size]
4703                            <1> ;     jnb   short swpin_snp_err
4704                            <1> ;
4705                            <1> ;     push  esi
4706                            <1> ;     push  ebx
4707                            <1> ;     push  ecx
4708                            <1> ;     mov   esi, [swp_drv]
4709                            <1> ;     mov   ecx, PAGE_SIZE / LOGIC_SECT_SIZE  ; 8 !
4710                            <1> ;           ; Note: Even if corresponding physical disk's sector
4711                            <1> ;           ; size different than 512 bytes, logical disk sector
4712                            <1> ;           ; size is 512 bytes and disk reading procedure
4713                            <1> ;           ; will be performed for reading 4096 bytes
4714                            <1> ;           ; (2*2048, 8*512).
4715                            <1> ;     ; ESI = Logical disk description table address
4716                            <1> ;     ; EBX = Memory page (buffer) address (physical!)
4717                            <1> ;     ; EAX = Sector adress (offset address, logical sector number)
4718                            <1> ;     ; ECX = Sector count ; 8 sectors
4719                            <1> ;     push  eax
4720                            <1> ;     call  logical_disk_read
4721                            <1> ;     pop   eax
4722                            <1> ;     jnc   short swpin_read_ok
4723                            <1> ;     ;
4724                            <1> ;     mov   eax, SWP_DISK_READ_ERR ; drive not ready or read error
4725                            <1> ;     mov   [u.error], eax
4726                            <1> ;     jmp   short swpin_retn
4727                            <1> ;     ;
4728                            <1> ;swpin_read_ok:
4729                            <1> ;     ; EAX = Offset address (logical sector number)
4730                            <1> ;     call  unlink_swap_block  ; Deallocate swap block
4731                            <1> ;     ;
4732                            <1> ;     ; EBX = Memory page (buffer) address (physical!)
4733                            <1> ;     ; 20/07/2015
4734                            <1> ;     mov   ebx, ebp ; virtual address (page fault address)
4735                            <1> ;       and     bx, ~PAGE_OFF ; ~0FFFh ; reset bits, 0 to 11
4736                            <1> ;     mov   bl, [u.uno] ; current process number
4737                            <1> ;     ; EBX = Virtual (Linear) address & process number combination
4738                            <1> ;     call  swap_queue_shift
4739                            <1> ;     ; eax = 0 ; 10/06/2016 (if ebx input > 0, eax output = 0)
4740                            <1> ;     ;sub   eax, eax  ; 0 ; Error Code = 0  (no error)
4741                            <1> ;     ; zf = 1
4742                            <1> ;swpin_retn:
4743                            <1> ;     pop   ecx
4744                            <1> ;     pop   ebx
4745                            <1> ;     pop   esi
4746                            <1> ;     retn
4747                            <1> ;
4748                            <1> ;swpin_dnp_err:
4749                            <1> ;     mov   eax, SWP_DISK_NOT_PRESENT_ERR
4750                            <1> ;swpin_err_retn:
4751                            <1> ;     mov   [u.error], eax
4752                            <1> ;     stc
4753                            <1> ;     retn
4754                            <1> ;
4755                            <1> ;swpin_snp_err:
4756                            <1> ;     mov   eax, SWP_SECTOR_NOT_PRESENT_ERR
4757                            <1> ;     jmp   short swpin_err_retn
4758                            <1>
4759                            <1> ; 17/04/2021
4760                            <1> ; ('swap_out' procedure call is disabled as temporary)
4761                            <1>
4762                            <1> swap_out:
4763                            <1>       ; 10/06/2016
4764                            <1>       ; 07/06/2016
4765                            <1>         ; 23/05/2016
4766                            <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
4767                            <1>       ; 24/10/2014 - 31/08/2015 (Retro UNIX 386 v1)
4768                            <1>       ;
4769                            <1>       ; INPUT ->
4770                            <1>       ;     none
4771                            <1>       ;
4772                            <1>       ; OUTPUT ->
4773                            <1>       ;     EAX = Physical page address (which is swapped out
4774                            <1>       ;           for allocating a new page)
4775                            <1>       ;     CF = 1 -> swap disk writing error (disk/file not present
4776                            <1>       ;               or sector not present or drive not ready
4777                            <1>       ;          EAX = Error code
4778                            <1>       ;          [u.error] = EAX
4779                            <1>       ;                  = The last error code for the process
4780                            <1>       ;                    (will be reset after returning to user)
4781                            <1>       ;
4782                            <1>       ; Modified Registers -> none (except EAX)
4783                            <1>       ;
4784                            <1>
4785                            <1> ;     cmp   word [swpq_count], 1
4786                            <1> ;     jc    swpout_im_err ; 'insufficient memory'
4787                            <1> ;
4788                            <1> ;     ;cmp  dword [swp_drv], 1
4789                            <1> ;     ;jc   short swpout_dnp_err ; 'swap disk/file not present'
4790                            <1> ;
4791                            <1> ;     cmp   dword [swpd_free], 1
4792                            <1> ;     jc    swpout_nfspc_err ; 'no free space on swap disk'
4793                            <1> ;
4794                            <1> ;     push  ebx ; *
4795                            <1> ;swpout_1:
4796                            <1> ;     ; 10/06/2016
4797                            <1> ;     xor   ebx, ebx ; shift the queue and return a PTE value
4798                            <1> ;     call  swap_queue_shift
```

```
4799                    <1> ;       and     eax, eax        ; 0 = empty queue (improper entries)
4800                    <1> ;       jz      swpout_npts_err ; There is not any proper PTE
4801                    <1> ;                               ; pointer in the swap queue
4802                    <1> ;       ; EAX = PTE value of the page
4803                    <1> ;       ; EBX = PTE address of the page
4804                    <1> ;       and     ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
4805                    <1> ;       ;
4806                    <1> ;       ; 07/06/2016
4807                    <1> ;       ; 19/05/2016
4808                    <1> ;       ; check this page is in timer events or not
4809                    <1> ;
4810                    <1> ;swpout_timer_page_0:
4811                    <1> ;       push    edx ; **
4812                    <1> ;
4813                    <1> ;       ; 07/06/2016
4814                    <1> ;       cmp     byte [timer_events], 0
4815                    <1> ;       jna     short swpout_2
4816                    <1> ;       ;
4817                    <1> ;       mov     dl, [timer_events]
4818                    <1> ;
4819                    <1> ;       push    ecx ; ***
4820                    <1> ;       push    ebx ; ****
4821                    <1> ;       mov     ebx, timer_set ; beginning address of timer event
4822                    <1> ;                              ; structures
4823                    <1> ;swpout_timer_page_1:
4824                    <1> ;       mov     cl, [ebx]
4825                    <1> ;       or      cl, cl ; 0 = free, >0 = process number
4826                    <1> ;       jz      short swpout_timer_page_3
4827                    <1> ;       mov     ecx, [ebx+12] ; response (signal return) address
4828                    <1> ;       and     cx, PTE_A_CLEAR ; clear offset part (right 12 bits)
4829                    <1> ;                               ; of the response byte address, to
4830                    <1> ;                               ; get beginning of the page address)
4831                    <1> ;       cmp     eax, ecx
4832                    <1> ;       jne     short swpout_timer_page_2 ; not same page
4833                    <1> ;
4834                    <1> ;       ; !same page!
4835                    <1> ;       ;
4836                    <1> ;       ; NOTE: // 19/05/2016 // - TRDOS 386 feature only ! -
4837                    <1> ;       ; This page will be used by the kernel to put timer event
4838                    <1> ;       ; response (signal return) byte at the requested address;
4839                    <1> ;       ; in order to prevent a possible wrong write (while
4840                    <1> ;       ; this page is swapped out) on physical memory,
4841                    <1> ;       ; we must protect this page against to be swapped out!
4842                    <1> ;       ;
4843                    <1> ;       pop     ebx ; ****
4844                    <1> ;       pop     ecx ; ***
4845                    <1> ;       pop     edx ; **
4846                    <1> ;       jmp     short swpout_1      ; do not swap out this page !
4847                    <1> ;
4848                    <1> ;swpout_timer_page_2:
4849                    <1> ;       ; 07/06/2016
4850                    <1> ;       dec     dl
4851                    <1> ;       jz      short swpout_timer_page_4
4852                    <1> ;swpout_timer_page_3:
4853                    <1> ;       ;cmp    ebx, timer_set + 240 ; last timer event (15*16)
4854                    <1> ;       ;jnb    short swpout_timer_page_4
4855                    <1> ;       add     ebx, 16
4856                    <1> ;       jmp     short swpout_timer_page_1
4857                    <1> ;
4858                    <1> ;swpout_timer_page_4:
4859                    <1> ;       pop     ebx ; ****
4860                    <1> ;       pop     ecx ; ***
4861                    <1> ;swpout_2:
4862                    <1> ;       mov     edx, ebx            ; Page table entry address
4863                    <1> ;       mov     ebx, eax            ; Buffer (Page) Address
4864                    <1> ;       ;
4865                    <1> ;       call    link_swap_block
4866                    <1> ;       jnc     short swpout_3            ; It may not be needed here
4867                    <1> ;                                   ; because [swpd_free] value
4868                    <1> ;                                   ; was checked at the beginging.
4869                    <1> ;       pop     edx ; **
4870                    <1> ;       pop     ebx ; *
4871                    <1> ;       jmp     short swpout_nfspc_err
4872                    <1> ;swpout_3:
4873                    <1> ;       test    eax, 80000000h ; test bit 31 (this may not be needed!)
4874                    <1> ;       jnz     short swpout_nfspc_err  ; 10/06/2016 (bit 31 = 1 !)
4875                    <1> ;       ;
4876                    <1> ;       push    esi ; **
4877                    <1> ;       push    ecx ; ***
4878                    <1> ;       push    eax ; sector address ; (31 bit !, bit 31 = 0)
4879                    <1> ;       mov     esi, [swp_drv]
4880                    <1> ;       mov     ecx, PAGE_SIZE / LOGIC_SECT_SIZE  ; 8 !
4881                    <1> ;               ; Note: Even if corresponding physical disk's sector
4882                    <1> ;               ; size different than 512 bytes, logical disk sector
4883                    <1> ;               ; size is 512 bytes and disk writing procedure
4884                    <1> ;               ; will be performed for writing 4096 bytes
4885                    <1> ;               ; (2*2048, 8*512).
4886                    <1> ;       ; ESI = Logical disk description table address
4887                    <1> ;       ; EBX = Buffer (Page) address
4888                    <1> ;       ; EAX = Sector adress (offset address, logical sector number)
4889                    <1> ;       ; ECX = Sector count ; 8 sectors
4890                    <1> ;       ; edx = PTE address
4891                    <1> ;       call    logical_disk_write
4892                    <1> ;       ; edx = PTE address
4893                    <1> ;       pop     ecx ; sector address
4894                    <1> ;       jnc     short swpout_write_ok
4895                    <1> ;       ;
4896                    <1> ;       ;; call      unlink_swap_block ; this block must be left as 'in use'
4897                    <1> ;swpout_dw_err:
4898                    <1> ;       mov     eax, SWP_DISK_WRITE_ERR ; drive not ready or write error
4899                    <1> ;       mov     [u.error], eax
4900                    <1> ;       jmp     short swpout_retn
4901                    <1> ;       ;
4902                    <1> ;swpout_write_ok:
4903                    <1> ;       ; EBX = Buffer (page) address
```

```
4904                    <1> ;       ; EDX = Page Table Entry address
4905                    <1> ;       ; ECX = Swap disk sector (file block) address (31 bit)
4906                    <1> ;       shl     ecx, 1  ; 31 bit sector address from bit 1 to bit 31
4907                    <1> ;       mov     [edx], ecx
4908                    <1> ;               ; bit 0 = 0 (swapped page)
4909                    <1> ;       mov     eax, ebx
4910                    <1> ;swpout_retn:
4911                    <1> ;       pop     ecx ; ***
4912                    <1> ;       pop     esi ; **
4913                    <1> ;       pop     ebx ; *
4914                    <1> ;       retn
4915                    <1> ;
4916                    <1> ;;swpout_dnp_err:
4917                    <1> ;;      mov     eax, SWP_DISK_NOT_PRESENT_ERR ; disk not present
4918                    <1> ;;      jmp     short swpout_err_retn
4919                    <1> ;swpout_nfspc_err:
4920                    <1> ;       mov     eax, SWP_NO_FREE_SPACE_ERR ; no free space
4921                    <1> ;swpout_err_retn:
4922                    <1> ;       mov     [u.error], eax
4923                    <1> ;       ;stc
4924                    <1> ;       retn
4925                    <1> ;swpout_npts_err:
4926                    <1> ;       mov     eax, SWP_NO_PAGE_TO_SWAP_ERR
4927                    <1> ;       pop     ebx
4928                    <1> ;       jmp     short swpout_err_retn
4929                    <1> ;swpout_im_err:
4930                    <1> ;       mov     eax, ERR_MINOR_IM ; insufficient (out of) memory
4931                    <1> ;       jmp     short swpout_err_retn
4932                    <1>
4933                    <1> ; 17/04/2021
4934                    <1> ; ('swap_queue_shift' procedure call is disabled as temporary)
4935                    <1>
4936                    <1> swap_queue_shift:
4937                    <1>       ; 26/03/2017
4938                    <1>       ; 10/06/2016
4939                    <1>       ; 09/06/2016 - TRDOS 386 (TRDOS v2.0)
4940                    <1>       ; 23/10/2014 - 20/07/2015 (Retro UNIX 386 v1)
4941                    <1>       ;
4942                    <1>       ; INPUT ->
4943                    <1>       ;     EBX = Virtual (linear) address (bit 12 to 31)
4944                    <1>       ;           and process number combination (bit 0 to 11)
4945                    <1>       ;     EBX = 0 -> shift/drop from the head (offset 0)
4946                    <1>       ;
4947                    <1>       ; OUTPUT ->
4948                    <1>       ;     If EBX input > 0
4949                    <1>       ;        the queue will be shifted 4 bytes (dword),
4950                    <1>       ;        from the tail to the head, up to entry offset
4951                    <1>       ;        which points to EBX input value or nothing
4952                    <1>       ;        to do if EBX value is not found on the queue.
4953                    <1>       ;        (The entry -with EBX value- will be removed
4954                    <1>       ;        from the queue if it is found.)
4955                    <1>       ;
4956                    <1>       ;        EAX = 0
4957                    <1>       ;
4958                    <1>       ;     If EBX input = 0
4959                    <1>       ;        the queue will be shifted 4 bytes (dword),
4960                    <1>       ;        from the tail to the head, if the PTE address
4961                    <1>       ;        which is pointed in head of the queue is marked
4962                    <1>       ;        as "accessed" or it is marked as "non present".
4963                    <1>       ;        (If "accessed" flag of the PTE -which is pointed
4964                    <1>       ;        in the head- is set -to 1-, it will be reset
4965                    <1>       ;        -to 0- and then, the queue will be rotated
4966                    <1>       ;        -without dropping pointer of the PTE from
4967                    <1>       ;        the queue- for 4 bytes on head to tail direction.
4968                    <1>       ;        Pointer in the head will be moved into the tail,
4969                    <1>       ;        other PTEs will be shifted on head direction.)
4970                    <1>       ;
4971                    <1>       ;        Swap queue will be shifted up to the first
4972                    <1>       ;        'present' or 'non accessed' page will be found
4973                    <1>       ;        (as pointed) on the queue head (then it will be
4974                    <1>       ;          removed/dropped from the queue).
4975                    <1>       ;
4976                    <1>       ;        EAX (> 0) = PTE value of the page which is
4977                    <1>       ;             (it's pointer -virtual address-) dropped
4978                    <1>       ;             (removed) from swap queue.
4979                    <1>       ;        EBX = PTE address of the page (if EAX > 0)
4980                    <1>       ;              which is (it's pointer -virtual address-)
4981                    <1>       ;             dropped (removed) from swap queue.
4982                    <1>       ;
4983                    <1>       ;        EAX = 0 -> empty swap queue !
4984                    <1>       ;
4985                    <1>       ; Modified Registers -> EAX, EBX
4986                    <1>       ;
4987                    <1> ;     movzx   eax, word [swpq_count]  ; Max. 1024
4988                    <1> ;     and     ax, ax
4989                    <1> ;     jz      short swpqs_retn
4990                    <1> ;     push    edi
4991                    <1> ;     push    esi
4992                    <1> ;     push    ecx
4993                    <1> ;     mov     esi, swap_queue
4994                    <1> ;     mov     ecx, eax
4995                    <1> ;     or      ebx, ebx
4996                    <1> ;     jz      short swpqs_7
4997                    <1> ;swpqs_1:
4998                    <1> ;     lodsd
4999                    <1> ;     cmp     eax, ebx
5000                    <1> ;     je      short swpqs_2
5001                    <1> ;     loop    swpqs_1
5002                    <1> ;     ; 10/06/2016
5003                    <1> ;     sub     eax, eax
5004                    <1> ;     jmp     short swpqs_6
5005                    <1> ;swpqs_2:
5006                    <1> ;     mov     edi, esi
5007                    <1> ;     sub     edi, 4
5008                    <1> ;swpqs_3:
```

```
5009                    <1> ;       dec     word [swpq_count]
5010                    <1> ;       jz      short swpqs_5
5011                    <1> ;swpqs_4:
5012                    <1> ;       dec     ecx
5013                    <1> ;       rep     movsd  ; shift up (to the head)
5014                    <1> ;swpqs_5:
5015                    <1> ;       xor     eax, eax
5016                    <1> ;       mov     [edi], eax
5017                    <1> ;swpqs_6:
5018                    <1> ;       pop     ecx
5019                    <1> ;       pop     esi
5020                    <1> ;       pop     edi
5021                    <1> ;swpqs_retn:
5022                    <1> ;       retn
5023                    <1> ;swpqs_7:
5024                    <1> ;       mov     edi, esi ; head
5025                    <1> ;       lodsd
5026                    <1> ;       ; 20/07/2015
5027                    <1> ;       mov     ebx, eax
5028                    <1> ;       and     ebx, ~PAGE_OFF ; ~0FFFh
5029                    <1> ;                       ; ebx = virtual address (at page boundary)
5030                    <1> ;       and     eax, PAGE_OFF ; 0FFFh
5031                    <1> ;                       ; ax = process number (1 to 4095)
5032                    <1> ;       cmp     al, [u.uno]
5033                    <1> ;               ; Max. 16 (nproc) processes for Retro UNIX 386 v1
5034                    <1> ;       jne     short swpqs_8
5035                    <1> ;       mov     eax, [u.pgdir]
5036                    <1> ;       jmp     short swpqs_9
5037                    <1> ;swpqs_8:
5038                    <1> ;       ; 09/06/2016
5039                    <1> ;       cmp     byte [eax+p.stat-1], 0
5040                    <1> ;       jna     short swpqs_3     ; free (or terminated) process
5041                    <1> ;       cmp     byte [eax+p.stat-1], 2 ; waiting
5042                    <1> ;       ja      short swpqs_3         ; zombie (3) or undefined ?
5043                    <1> ;
5044                    <1> ;       ;shl    ax, 2
5045                    <1> ;       shl     al, 2
5046                    <1> ;       mov     eax, [eax+p.upage-4]
5047                    <1> ;       or      eax, eax
5048                    <1> ;       jz      short swpqs_3 ; invalid upage
5049                    <1> ;       add     eax, u.pgdir - user
5050                    <1> ;                       ; u.pgdir value for the process
5051                    <1> ;                       ; is in [eax]
5052                    <1> ;       mov     eax, [eax]
5053                    <1> ;       and     eax, eax
5054                    <1> ;       jz      short swpqs_3 ; invalid page directory
5055                    <1> ;swpqs_9:
5056                    <1> ;       push    edx
5057                    <1> ;       ; eax = page directory
5058                    <1> ;       ; ebx = virtual address
5059                    <1> ;       call    get_pte
5060                    <1> ;       mov     ebx, edx     ; PTE address
5061                    <1> ;       pop     edx
5062                    <1> ;       ; 10/06/2016
5063                    <1> ;       jc      short swpqs_13 ; empty PDE
5064                    <1> ;       ; EAX = PTE value
5065                    <1> ;       test    al, PTE_A_PRESENT ; bit 0 = 1
5066                    <1> ;       jz      short swpqs_13  ; Drop non-present page
5067                    <1> ;                               ; from the queue (head)
5068                    <1> ;       test    al, PTE_A_WRITE     ; bit 1 = 0 (read only)
5069                    <1> ;       jz      short swpqs_13  ; Drop read only page
5070                    <1> ;                               ; from the queue (head)
5071                    <1> ;       ;test   al, PTE_A_ACCESS ; bit 5 = 1 (Accessed)
5072                    <1> ;       ;jnz    short swpqs_11  ; present
5073                    <1> ;                               ; accessed page
5074                    <1> ;       btr     eax, PTE_A_ACCESS_BIT ; reset 'accessed' bit
5075                    <1> ;       jc      short swpqs_11  ; accessed page
5076                    <1> ;
5077                    <1> ;       dec     ecx
5078                    <1> ;       mov     [swpq_count], cx
5079                    <1> ;       jz      short swpqs_10
5080                    <1> ;               ; esi = head + 4
5081                    <1> ;               ; edi = head
5082                    <1> ;       rep     movsd   ; n = 1 to k-1, [n - 1] = [n]
5083                    <1> ;swpqs_10:
5084                    <1> ;       mov     [edi], ecx ; 0
5085                    <1> ;       jmp     short swpqs_6 ; 26/03/2017
5086                    <1> ;
5087                    <1> ;swpqs_11:
5088                    <1> ;       mov     [ebx], eax     ; save changed attribute
5089                    <1> ;       ; Rotation (head -> tail)
5090                    <1> ;       dec     ecx     ; entry count -> last entry number
5091                    <1> ;       jz      short swpqs_10
5092                    <1> ;               ; esi = head + 4
5093                    <1> ;               ; edi = head
5094                    <1> ;       mov     eax, [edi] ; 20/07/2015
5095                    <1> ;       rep     movsd   ; n = 1 to k-1, [n - 1] = [n]
5096                    <1> ;       mov     [edi], eax ; head -> tail ; [k] = [1]
5097                    <1> ;
5098                    <1> ;       mov     cx, [swpq_count]
5099                    <1> ;
5100                    <1> ;swpqs_12:
5101                    <1> ;       mov     esi, swap_queue ; head
5102                    <1> ;       jmp     swpqs_7
5103                    <1> ;
5104                    <1> ;swpqs_13:
5105                    <1> ;       dec     ecx
5106                    <1> ;       mov     [swpq_count], cx
5107                    <1> ;       jz      swpqs_5
5108                    <1> ;       jmp     short swpqs_12
5109                    <1>
5110                    <1> ; 17/04/2021
5111                    <1> ; ('add_to_swp_queue' procedure call is disabled as temporary)
5112                    <1>
5113                    <1> add_to_swap_queue:
```

```
5114                            <1>       ; 20/02/2017
5115                            <1>       ; 20/07/2015
5116                            <1>       ; 24/10/2014 (Retro UNIX 386 v1 - beginning)
5117                            <1>       ;
5118                            <1>       ; Adds new page to swap queue
5119                            <1>       ; (page directories and page tables must not be added
5120                            <1>       ; to swap queue)
5121                            <1>       ;
5122                            <1>       ; INPUT ->
5123                            <1>       ;     EBX = Linear (Virtual) addr for current process
5124                            <1>       ;     [u.uno]
5125                            <1>       ;     20/02/2017
5126                            <1>       ;     (Linear address = CORE + user's virtual address)
5127                            <1>       ;
5128                            <1>       ; OUTPUT ->
5129                            <1>       ;     EAX = [swpq_count]
5130                            <1>       ;           (after the PTE has been added)
5131                            <1>       ;     EAX = 0 -> Swap queue is full, (1024 entries)
5132                            <1>       ;           the PTE could not be added.
5133                            <1>       ;
5134                            <1>       ; Modified Registers -> EAX
5135                            <1>       ;
5136                            <1> ;     push   ebx
5137                            <1> ;     and   bx, ~PAGE_OFF ; ~0FFFh ; reset bits, 0 to 11
5138                            <1> ;     mov    bl, [u.uno] ; current process number
5139                            <1> ;     call   swap_queue_shift ; drop from the queue if
5140                            <1> ;                       ; it is already on the queue
5141                            <1> ;                      ; then add it to the tail of the queue
5142                            <1> ;     movzx  eax, word [swpq_count]
5143                            <1> ;     cmp    ax, 1024
5144                            <1> ;     jb     short atsq_1
5145                            <1> ;     sub    ax, ax
5146                            <1> ;     pop    ebx
5147                            <1> ;     retn
5148                            <1> ;atsq_1:
5149                            <1> ;     push   esi
5150                            <1> ;     mov    esi, swap_queue
5151                            <1> ;     and    ax, ax
5152                            <1> ;     jz     short atsq_2
5153                            <1> ;     shl    ax, 2  ; convert to offset
5154                            <1> ;     add    esi, eax
5155                            <1> ;     shr    ax, 2
5156                            <1> ;atsq_2:
5157                            <1> ;     inc    ax
5158                            <1> ;     mov    [esi], ebx ; Virtual address + [u.uno] combination
5159                            <1> ;     mov    [swpq_count], ax
5160                            <1> ;     pop    esi
5161                            <1> ;     pop    ebx
5162                            <1> ;     retn
5163                            <1>
5164                            <1> ; 17/04/2021
5165                            <1> ; ('unlink_swap_block' procedure call is disabled as temporary)
5166                            <1>
5167                            <1> unlink_swap_block:
5168                            <1>       ; 15/09/2015
5169                            <1>       ; 30/04/2015
5170                            <1>       ; 18/04/2015
5171                            <1>       ; 24/10/2014 (Retro UNIX 386 v1 - beginning)
5172                            <1>       ;
5173                            <1>       ; INPUT ->
5174                            <1>       ;     EAX = swap disk/file offset address
5175                            <1>       ;           (bit 1 to bit 31)
5176                            <1>       ; OUTPUT ->
5177                            <1>       ;     [swpd_free] is increased
5178                            <1>       ;     (corresponding SWAP DISK ALLOC. TABLE bit is SET)
5179                            <1>       ;
5180                            <1>       ; Modified Registers -> EAX
5181                            <1>       ;
5182                            <1> ;     push   ebx
5183                            <1> ;     push   edx
5184                            <1> ;     ;
5185                            <1> ;     shr    eax, SECTOR_SHIFT+1  ;3+1 ; shift sector address to
5186                            <1> ;                                  ; 3 bits right
5187                            <1> ;                                  ; to get swap block/page number
5188                            <1> ;     mov    edx, eax
5189                            <1> ;     ; 15/09/2015
5190                            <1> ;     shr    edx, 3           ; to get offset to S.A.T.
5191                            <1> ;                               ; (1 allocation bit = 1 page)
5192                            <1> ;                               ; (1 allocation bytes = 8 pages)
5193                            <1> ;     and    dl, 0FCh         ; clear lower 2 bits
5194                            <1> ;                               ; (to get 32 bit position)
5195                            <1> ;     ;
5196                            <1> ;     mov    ebx, swap_alloc_table ; Swap Allocation Table address
5197                            <1> ;     add    ebx, edx
5198                            <1> ;     and    eax, 1Fh         ; lower 5 bits only
5199                            <1> ;                               ; (allocation bit position)
5200                            <1> ;     cmp    eax, [swpd_next]     ; is the new free block addr. lower
5201                            <1> ;                               ; than the address in 'swpd_next' ?
5202                            <1> ;                               ; (next/first free block value)
5203                            <1> ;     jnb    short uswpbl_1          ; no
5204                            <1> ;     mov    [swpd_next], eax     ; yes
5205                            <1> ;uswpbl_1:
5206                            <1> ;     bts    [ebx], eax       ; unlink/release/deallocate block
5207                            <1> ;                               ; set relevant bit to 1.
5208                            <1> ;                               ; set CF to the previous bit value
5209                            <1> ;     cmc                      ; complement carry flag
5210                            <1> ;     jc     short uswpbl_2    ; do not increase swfd_free count
5211                            <1> ;                               ; if the block is already deallocated
5212                            <1> ;                               ; before.
5213                            <1> ;     inc   dword [swpd_free]
5214                            <1> ;uswpbl_2:
5215                            <1> ;     pop    edx
5216                            <1> ;     pop    ebx
5217                            <1> ;     retn
5218                            <1>
```

```
                        <1> ; 17/04/2021
                        <1> ; ('ink_swap_block' procedure call is disabled as temporary)
                        <1>
                        <1> link_swap_block:
                        <1>       ; 01/07/2015
                        <1>       ; 18/04/2015
                        <1>       ; 24/10/2014 (Retro UNIX 386 v1 - beginning)
                        <1>       ;
                        <1>       ; INPUT -> none
                        <1>       ;
                        <1>       ; OUTPUT ->
                        <1>       ;     EAX = OFFSET ADDRESS OF THE ALLOCATED BLOCK (4096 bytes)
                        <1>       ;           in sectors (corresponding
                        <1>       ;           SWAP DISK ALLOCATION TABLE bit is RESET)
                        <1>       ;
                        <1>       ;     CF = 1 and EAX = 0
                        <1>       ;               if there is not a free block to be allocated
                        <1>       ;
                        <1>       ; Modified Registers -> none (except EAX)
                        <1>       ;
                        <1>
                        <1> ;     ;mov   eax, [swpd_free]
                        <1> ;     ;and   eax, eax
                        <1> ;     ;jz    short out_of_swpspc
                        <1> ;     ;
                        <1> ;     push   ebx
                        <1> ;     push   ecx
                        <1> ;     ;
                        <1> ;     mov    ebx, swap_alloc_table ; Swap Allocation Table offset
                        <1> ;     mov    ecx, ebx
                        <1> ;     add    ebx, [swpd_next] ; Free block searching starts from here
                        <1> ;                             ; next_free_swap_block >> 5
                        <1> ;     add    ecx, [swpd_last] ; Free block searching ends here
                        <1> ;                             ; (total_swap_blocks - 1) >> 5
                        <1> ;lswbl_scan:
                        <1> ;     cmp    ebx, ecx
                        <1> ;     ja     short lswbl_notfound
                        <1> ;     ;
                        <1> ;     bsf    eax, [ebx] ; Scans source operand for first bit set (1).
                        <1> ;                       ; Clears ZF if a bit is found set (1) and
                        <1> ;                       ; loads the destination with an index to
                        <1> ;                       ; first set bit. (0 -> 31)
                        <1> ;                       ; Sets ZF to 1 if no bits are found set.
                        <1> ;     ; 01/07/2015
                        <1> ;     jnz    short lswbl_found ; ZF = 0 -> a free block has been found
                        <1> ;                     ;
                        <1> ;                     ; NOTE:  a Swap Disk Allocation Table bit
                        <1> ;                     ;       with value of 1 means
                        <1> ;                     ;       the corresponding page is free
                        <1> ;                     ;       (Retro UNIX 386 v1 feaure only!)
                        <1> ;     add    ebx, 4
                        <1> ;                     ; We return back for searching next page block
                        <1> ;                     ; NOTE: [swpd_free] is not ZERO; so,
                        <1> ;                     ;     we always will find at least 1 free block here.
                        <1> ;     jmp          short lswbl_scan
                        <1> ;     ;
                        <1> ;lswbl_notfound:
                        <1> ;     sub    ecx, swap_alloc_table
                        <1> ;     mov    [swpd_next], ecx ; next/first free page = last page
                        <1> ;                             ; (unlink_swap_block procedure will change it)
                        <1> ;     xor    eax, eax
                        <1> ;     mov    [swpd_free], eax
                        <1> ;     stc
                        <1> ;lswbl_ok:
                        <1> ;     pop    ecx
                        <1> ;     pop    ebx
                        <1> ;     retn
                        <1> ;     ;
                        <1> ;;out_of_swpspc:
                        <1> ;;    stc
                        <1> ;;    retn
                        <1> ;
                        <1> ;lswbl_found:
                        <1> ;     mov    ecx, ebx
                        <1> ;     sub    ecx, swap_alloc_table
                        <1> ;     mov    [swpd_next], ecx ; Set first free block searching start
                        <1> ;                             ; address/offset (to the next)
                        <1> ;     dec   dword [swpd_free] ; 1 block has been allocated (X = X-1)
                        <1> ;     ;
                        <1> ;     btr    [ebx], eax    ; The destination bit indexed by the source value
                        <1> ;                           ; is copied into the Carry Flag and then cleared
                        <1> ;                           ; in the destination.
                        <1> ;                           ;
                        <1> ;                           ; Reset the bit which is corresponding to the
                        <1> ;                           ; (just) allocated block.
                        <1> ;     shl    ecx, 5        ; (block offset * 32) + block index
                        <1> ;     add    eax, ecx      ; = block number
                        <1> ;     shl    eax, SECTOR_SHIFT ; 3, sector (offset) address of the block
                        <1> ;                           ; 1 block =  8 sectors
                        <1> ;     ;
                        <1> ;     ; EAX = offset address of swap disk/file sector (beginning of the block)
                        <1> ;     ;
                        <1> ;     ; NOTE: The relevant page table entry will be updated
                        <1> ;     ;       according to this EAX value...
                        <1> ;     ;
                        <1> ;     jmp    short lswbl_ok
                        <1>
                        <1> ; 17/04/2021
                        <1> ; ('logical_disk_read' procedure call is disabled as temporary)
                        <1>
                        <1> logical_disk_read:
                        <1>       ; 20/07/2015
                        <1>       ; 09/03/2015 (temporary code here)
                        <1>       ;
                        <1>       ; INPUT ->
```

```
5324                                    <1>       ;      ESI = Logical disk description table address
5325                                    <1>       ;      EBX = Memory page (buffer) address (physical!)
5326                                    <1>       ;      EAX = Sector adress (offset address, logical sector number)
5327                                    <1>       ;      ECX = Sector count
5328                                    <1>       ;
5329                                    <1>       ;
5330                                    <1> ;     retn
5331                                    <1>
5332                                    <1> ; 17/04/2021
5333                                    <1> ; ('logical_disk_write' procedure call is disabled as temporary)
5334                                    <1>
5335                                    <1> logical_disk_write:
5336                                    <1>       ; 20/07/2015
5337                                    <1>       ; 09/03/2015 (temporary code here)
5338                                    <1>       ;
5339                                    <1>       ; INPUT ->
5340                                    <1>       ;      ESI = Logical disk description table address
5341                                    <1>       ;      EBX = Memory page (buffer) address (physical!)
5342                                    <1>       ;      EAX = Sector adress (offset address, logical sector number)
5343                                    <1>       ;      ECX = Sector count
5344                                    <1>       ;
5345                                    <1> ;     retn
5346                                    <1>
5347                                    <1> get_physical_addr:
5348                                    <1>       ; 17/04/2021 - TRDOS 386 v2.0.4
5349                                    <1>       ;      (temporary modifications)
5350                                    <1>       ;
5351                                    <1>       ; 26/03/2017
5352                                    <1>       ; 20/02/2017
5353                                    <1>       ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
5354                                    <1>       ; 18/10/2015
5355                                    <1>       ; 29/07/2015
5356                                    <1>       ; 20/07/2015
5357                                    <1>       ; 04/06/2015
5358                                    <1>       ; 20/05/2015
5359                                    <1>       ; 28/04/2015
5360                                    <1>       ; 18/04/2015
5361                                    <1>       ; Get physical address
5362                                    <1>       ;     (allocates a new page for user if it is not present)
5363                                    <1>       ;
5364                                    <1>       ; (This subroutine is needed for mapping user's virtual
5365                                    <1>       ; (buffer) address to physical address (of the buffer).)
5366                                    <1>       ; ('sys write', 'sys read' system calls...)
5367                                    <1>       ;
5368                                    <1>       ; INPUT ->
5369                                    <1>       ;     EBX = virtual address
5370                                    <1>       ;     u.pgdir = page directory (physical) address
5371                                    <1>       ;
5372                                    <1>       ; OUTPUT ->
5373                                    <1>       ;     EAX = physical address
5374                                    <1>       ;     EBX = linear address
5375                                    <1>       ;     EDX = physical address of the page frame
5376                                    <1>       ;           (with attribute bits)
5377                                    <1>       ;     ECX = byte count within the page frame
5378                                    <1>       ;
5379                                    <1>       ; Modified Registers -> EAX, EBX, ECX, EDX
5380                                    <1>       ;
5381 00005E62 81C300004000             <1>       add   ebx, CORE ; 18/10/2015
5382                                    <1> get_physical_addr_x: ; 27/05/2016
5383 00005E68 A1[B8030300]             <1>       mov   eax, [u.pgdir]
5384 00005E6D E815FDFFFF               <1>       call  get_pte
5385                                    <1>             ; EDX = Page table entry address (if CF=0)
5386                                    <1>             ;       Page directory entry address (if CF=1)
5387                                    <1>             ;       (Bit 0 value is 0 if PT is not present)
5388                                    <1>             ; EAX = Page table entry value (page address)
5389                                    <1>             ;     CF = 1 -> PDE not present or invalid ?
5390 00005E72 731C                     <1>       jnc   short gpa_1
5391                                    <1>       ;
5392 00005E74 E8FCFBFFFF               <1>       call  allocate_page
5393 00005E79 723F                     <1>       jc    short gpa_im_err  ; 'insufficient memory' error
5394                                    <1> gpa_0:
5395 00005E7B E866FCFFFF               <1>       call  clear_page
5396                                    <1>       ; EAX = Physical (base) address of the allocated (new) page
5397 00005E80 0C07                     <1>       or    al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER ; 4+2+1 = 7
5398                                    <1>                       ; lower 3 bits are used as U/S, R/W, P flags
5399                                    <1>                       ; (user, writable, present page)
5400 00005E82 8902                     <1>       mov   [edx], eax ; Let's put the new page directory entry here !
5401 00005E84 A1[B8030300]             <1>       mov   eax, [u.pgdir]
5402 00005E89 E8F9FCFFFF               <1>       call  get_pte
5403 00005E8E 722A                     <1>       jc    short gpa_im_err ; 'insufficient memory' error
5404                                    <1> gpa_1:
5405                                    <1>       ; EAX = PTE value, EDX = PTE address
5406 00005E90 A801                     <1>       test  al, PTE_A_PRESENT
5407 00005E92 7516                     <1>       jnz   short gpa_3 ; 26/03/2017
5408 00005E94 09C0                     <1>       or    eax, eax
5409 00005E96 7446                     <1>       jz    short gpa_7  ; Allocate a new page
5410                                    <1>
5411                                    <1> ; 17/04/2021 (TRDOS 386 v2.0.4)
5412                                    <1> ; ('reload_page' procedure call is disabled as temporary)
5413 00005E98 EB20                     <1>       jmp   short gpa_im_err  ; temporary !
5414                                    <1>
5415                                    <1>       ; 20/07/2015
5416                                    <1> ;     push  ebp
5417                                    <1> ;     mov   ebp, ebx ; virtual (linear) address
5418                                    <1> ;     ; reload swapped page
5419                                    <1> ;     call  reload_page ; 28/04/2015
5420                                    <1> ;     pop   ebp
5421                                    <1> ;     jc    short gpa_retn
5422                                    <1> gpa_2:
5423                                    <1>       ; 26/03/2017
5424                                    <1>       ; 20/02/2017
5425                                    <1>       ; If a page will contain a Signal Response Byte
5426                                    <1>       ; it must not be swapped out, because
5427                                    <1>       ; timer service or irq callback service
5428                                    <1>       ; will write a signal return/response byte
```

```
                                <1>       ; directly by using physical address of Signal
                                <1>       ; Response Byte.(Even if process is not running,
                                <1>       ; or it is running with swapped out pages.)
                                <1>       ;
                                <1>       ; 'no_page_swap' will be set by 'systimer' or
                                <1>       ; 'syscalbac' sistem functions/calls. (*)
                                <1>       ;
00005E9A 803D[468F0100]00       <1>       cmp   byte [no_page_swap], 0
00005EA1 761D                   <1>       jna   short gpa_4 ; this page can be swapped out
                                <1>       ; this page must not be swapped out
                                <1>       ; but 'no_page_swap' must be reset here
                                <1>       ; immediately for other callers (*)
                                <1>       ; (otherwise, swap queue would not be long enough)
00005EA3 E844000000             <1>       call  gpa_8 ; 26/03/2017
00005EA8 EB16                   <1>       jmp   short gpa_5
                                <1> gpa_3:
                                <1>       ; 26/03/2017
00005EAA 803D[468F0100]00       <1>       cmp   byte [no_page_swap], 0
00005EB1 7611                   <1>       jna   short gpa_6 ; this page can be swapped out
00005EB3 E834000000             <1>       call  gpa_8
00005EB8 EB0A                   <1>       jmp   short gpa_6
                                <1>
                                <1> gpa_im_err:
00005EBA B804000000             <1>       mov   eax, ERR_MINOR_IM ; Insufficient memory (minor) error!
                                <1>                               ; Major error = 0 (No protection fault)
00005EBF C3                     <1>       retn
                                <1> gpa_4:
                                <1> ; 17/04/2021 (TRDOS 386 v2.0.4)
                                <1> ; ('add_to_swap_queue' procedure call is disabled as temporary)
                                <1>
                                <1>       ; 20/07/2015
                                <1>       ; 20/05/2015
                                <1>       ; add this page to swap queue
                                <1> ;     push  eax
                                <1> ;     ; EBX = Linear (CORE+virtual) address ; 20/02/2017
                                <1> ;     call  add_to_swap_queue
                                <1> ;     pop   eax
                                <1> gpa_5:
                                <1>             ; PTE address in EDX
                                <1>             ; virtual address in EBX
                                <1>       ; EAX = memory page address
00005EC0 0C07                   <1>       or    al, PTE_A_PRESENT + PTE_A_USER + PTE_A_WRITE
                                <1>                               ; present flag, bit 0 = 1
                                <1>                               ; user flag, bit 2 = 1
                                <1>                               ; writable flag, bit 1 = 1
00005EC2 8902                   <1>       mov   [edx], eax  ; Update PTE value
                                <1> gpa_6:
                                <1>       ; 18/10/2015
00005EC4 89D9                   <1>       mov   ecx, ebx
00005EC6 81E1FF0F0000           <1>       and   ecx, PAGE_OFF
00005ECC 89C2                   <1>       mov   edx, eax
00005ECE 662500F0               <1>       and   ax, PTE_A_CLEAR
00005ED2 01C8                   <1>       add   eax, ecx
00005ED4 F7D9                   <1>       neg   ecx ; 1 -> -1 (0FFFFFFFFh), 4095 (0FFFh) -> -4095
00005ED6 81C100100000           <1>       add   ecx, PAGE_SIZE
00005EDC F8                     <1>       clc
                                <1> gpa_retn:
00005EDD C3                     <1>       retn
                                <1> gpa_7:
00005EDE E892FBFFFF             <1>       call  allocate_page
00005EE3 72D5                   <1>       jc    short gpa_im_err ; 'insufficient memory' error
00005EE5 E8FCFBFFFF             <1>       call  clear_page
00005EEA EBAE                   <1>       jmp   short gpa_2
                                <1>
                                <1> gpa_8: ; 26/03/2017
00005EEC C605[468F0100]00       <1>       mov   byte [no_page_swap], 0
                                <1>
00005EF3 C3                     <1>       retn  ; 17/04/2021 (temporary)
                                <1>
                                <1> ; 17/04/2021 (TRDOS 386 v2.0.4)
                                <1> ; ('swap_queue_shift' procedure call is disabled as temporary)
                                <1> ;
                                <1> ;     push  ebx
                                <1> ;     push  eax ; 26/03/2017
                                <1> ;       and     bx, ~PAGE_OFF ; ~0FFFh ; reset bits, 0 to 11
                                <1> ;     mov   bl, [u.uno] ; current process number
                                <1> ;     call  swap_queue_shift ; drop from the queue if
                                <1> ;                            ; it is already on the queue
                                <1> ;     pop   eax ; 26/03/2017
                                <1> ;     pop   ebx
                                <1> ;
                                <1> ;     retn
                                <1>
                                <1> ; 17/04/2021
                                <1> ; ('reload_page' procedure call is disabled as temporary)
                                <1>
                                <1> reload_page:
                                <1>       ; 20/07/2015
                                <1>       ; 28/04/2015 (Retro UNIX 386 v1 - beginning)
                                <1>       ;
                                <1>       ; Reload (Restore) swapped page at memory
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     EBP = Virtual (linear) memory address
                                <1>       ;     EAX = PTE value (swap disk sector address)
                                <1>       ;     (Swap disk sector address = bit 1 to bit 31 of EAX)
                                <1>       ; OUTPUT ->
                                <1>       ;     EAX = PHYSICAL (real/flat) ADDRESS OF RELOADED PAGE
                                <1>       ;
                                <1>       ;     CF = 1 and EAX = error code
                                <1>       ;
                                <1>       ; Modified Registers -> none (except EAX)
                                <1>       ;
                                <1> ;     shr   eax, 1   ; Convert PTE value to swap disk address
                                <1> ;     push  ebx      ;
```

```
5534                                <1> ;       mov     ebx, eax ; Swap disk (offset) address
5535                                <1> ;       call    allocate_page
5536                                <1> ;       jc      short rlp_im_err
5537                                <1> ;       xchg    eax, ebx
5538                                <1> ;       ; EBX = Physical memory (page) address
5539                                <1> ;       ; EAX = Swap disk (offset) address
5540                                <1> ;       ; EBP = Virtual (linear) memory address
5541                                <1> ;       call    swap_in
5542                                <1> ;       jc      short rlp_swp_err  ; (swap disk/file read error)
5543                                <1> ;       mov     eax, ebx
5544                                <1> ;rlp_retn:
5545                                <1> ;       pop     ebx
5546                                <1> ;       retn
5547                                <1> ;
5548                                <1> ;rlp_im_err:
5549                                <1> ;       mov     eax, ERR_MINOR_IM ; Insufficient memory (minor) error!
5550                                <1> ;                                 ; Major error = 0 (No protection fault)
5551                                <1> ;       jmp     short rlp_retn
5552                                <1> ;
5553                                <1> ;rlp_swp_err:
5554                                <1> ;       mov     eax, SWP_DISK_READ_ERR ; Swap disk read error !
5555                                <1> ;       jmp     short rlp_retn
5556                                <1>
5557                                <1> copy_page_dir:
5558                                <1>         ; 17/04/2021 (temporary modifications)
5559                                <1>         ; 19/09/2015
5560                                <1>         ; temporary - 07/09/2015
5561                                <1>         ; 07/09/2015 (Retro UNIX 386 v1 - beginning)
5562                                <1>         ;
5563                                <1>         ; INPUT ->
5564                                <1>         ;       [u.pgdir] = PHYSICAL (real/flat) ADDRESS of the parent's
5565                                <1>         ;                   page directory.
5566                                <1>         ; OUTPUT ->
5567                                <1>         ;       EAX =  PHYSICAL (real/flat) ADDRESS of the child's
5568                                <1>         ;              page directory.
5569                                <1>         ;       (New page directory with new page table entries.)
5570                                <1>         ;       (New page tables with read only copies of the parent's
5571                                <1>         ;       pages.)
5572                                <1>         ;       EAX = 0 -> Error (CF = 1)
5573                                <1>         ;
5574                                <1>         ; Modified Registers -> none (except EAX)
5575                                <1>         ;
5576 00005EF4 E87CFBFFFF           <1>         call    allocate_page
5577 00005EF9 723E                 <1>         jc      short cpd_err
5578                                <1>         ;
5579 00005EFB 55                   <1>         push    ebp ; 20/07/2015
5580 00005EFC 56                   <1>         push    esi
5581 00005EFD 57                   <1>         push    edi
5582 00005EFE 53                   <1>         push    ebx
5583 00005EFF 51                   <1>         push    ecx
5584 00005F00 8B35[B8030300]       <1>         mov     esi, [u.pgdir]
5585 00005F06 89C7                 <1>         mov     edi, eax
5586 00005F08 50                   <1>         push    eax ; save child's page directory address
5587                                <1>         ; copy PDE 0 from the parent's page dir to the child's page dir
5588                                <1>         ; (use same system space for all user page tables)
5589 00005F09 A5                   <1>         movsd
5590 00005F0A BD00004000           <1>         mov     ebp, 1024*4096 ; pass the 1st 4MB (system space)
5591 00005F0F B9FF030000           <1>         mov     ecx, (PAGE_SIZE / 4) - 1 ; 1023
5592                                <1> cpd_0:
5593 00005F14 AD                   <1>         lodsd
5594                                <1>         ;or     eax, eax
5595                                <1>           ;jnz      short cpd_1
5596 00005F15 A801                 <1>         test    al, PDE_A_PRESENT ;  bit 0 =  1
5597 00005F17 7508                 <1>         jnz     short cpd_1
5598                                <1>         ; (virtual address at the end of the page table)
5599 00005F19 81C500004000         <1>         add     ebp, 1024*4096 ; page size * PTE count
5600 00005F1F EB0F                 <1>         jmp     short cpd_2
5601                                <1> cpd_1:
5602 00005F21 662500F0             <1>         and     ax, PDE_A_CLEAR ; 0F000h ; clear attribute bits
5603 00005F25 89C3                 <1>         mov     ebx, eax
5604                                <1>         ; EBX = Parent's page table address
5605 00005F27 E81F000000           <1>         call    copy_page_table
5606 00005F2C 720C                 <1>         jc      short cpd_p_err
5607                                <1>         ; EAX = Child's page table address
5608 00005F2E 0C07                 <1>         or      al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER
5609                                <1>                         ; set bit 0, bit 1 and bit 2 to 1
5610                                <1>                         ; (present, writable, user)
5611                                <1> cpd_2:
5612 00005F30 AB                   <1>         stosd
5613 00005F31 E2E1                 <1>         loop    cpd_0
5614                                <1>         ;
5615 00005F33 58                   <1>         pop     eax  ; restore child's page directory address
5616                                <1> cpd_3:
5617 00005F34 59                   <1>         pop     ecx
5618 00005F35 5B                   <1>         pop     ebx
5619 00005F36 5F                   <1>         pop     edi
5620 00005F37 5E                   <1>         pop     esi
5621 00005F38 5D                   <1>         pop     ebp
5622                                <1> cpd_err:
5623 00005F39 C3                   <1>         retn
5624                                <1> cpd_p_err:
5625                                <1>         ; release the allocated pages missing (recover free space)
5626 00005F3A 58                   <1>         pop     eax  ; the new page directory address (physical)
5627 00005F3B 8B1D[B8030300]       <1>         mov     ebx, [u.pgdir] ; parent's page directory address
5628 00005F41 E85FFCFFFF           <1>         call    deallocate_page_dir
5629 00005F46 29C0                 <1>         sub     eax, eax ; 0
5630 00005F48 F9                   <1>         stc
5631 00005F49 EBE9                 <1>         jmp     short cpd_3
5632                                <1>
5633                                <1> copy_page_table:
5634                                <1>         ; 17/04/2021 (temporary modifications)
5635                                <1>         ; 19/09/2015
5636                                <1>         ; temporary - 07/09/2015
5637                                <1>         ; 07/09/2015 (Retro UNIX 386 v1 - beginning)
5638                                <1>         ;
```

```
5639                                    <1>       ; INPUT ->
5640                                    <1>       ;      EBX = PHYSICAL (real/flat) ADDRESS of the parent's page table.
5641                                    <1>       ;      EBP = page table entry index (from 'copy_page_dir')
5642                                    <1>       ; OUTPUT ->
5643                                    <1>       ;      EAX = PHYSICAL (real/flat) ADDRESS of the child's page table.
5644                                    <1>       ;      EBP = (recent) page table index (for 'add_to_swap_queue')
5645                                    <1>       ;      CF = 1 -> error
5646                                    <1>       ;
5647                                    <1>       ; Modified Registers -> EBP (except EAX)
5648                                    <1>       ;
5649 00005F4B E825FBFFFF              <1>       call   allocate_page
5650 00005F50 7244                    <1>       jc     short cpt_err
5651                                    <1>       ;
5652 00005F52 50                      <1>       push   eax ; *
5653                                    <1>       ;push  ebx
5654 00005F53 56                      <1>       push   esi
5655 00005F54 57                      <1>       push   edi
5656 00005F55 52                      <1>       push   edx
5657 00005F56 51                      <1>       push   ecx
5658                                    <1>       ;
5659 00005F57 89DE                    <1>       mov    esi, ebx
5660 00005F59 89C7                    <1>       mov    edi, eax
5661 00005F5B 89C2                    <1>       mov    edx, eax
5662 00005F5D 81C200100000            <1>       add    edx, PAGE_SIZE
5663                                    <1> cpt_0:
5664 00005F63 AD                      <1>       lodsd
5665 00005F64 A801                    <1>       test   al, PTE_A_PRESENT ;  bit 0 = 1
5666                                    <1>       ;jnz   short cpt_1 (*)
5667                                    <1>       ; 17/04/2021 (temporary (*)
5668                                    <1>       ;and   eax, eax (*)
5669 00005F66 741E                    <1>       jz     short cpt_2  ; 17/04/2021
5670                                    <1>
5671                                    <1> ; 17/04/2021
5672                                    <1> ; ('reload_page' procedure call is disabled as temporary)
5673                                    <1> ;
5674                                    <1> ;     ; ebp = virtual (linear) address of the memory page
5675                                    <1> ;     call   reload_page ; 28/04/2015
5676                                    <1> ;     jc     short cpt_p_err
5677                                    <1> cpt_1:
5678 00005F68 662500F0                <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
5679 00005F6C 89C1                    <1>       mov    ecx, eax
5680                                    <1>       ; Allocate a new page for the child process
5681 00005F6E E802FBFFFF              <1>       call   allocate_page
5682 00005F73 721C                    <1>       jc     short cpt_p_err
5683 00005F75 57                      <1>       push   edi
5684 00005F76 56                      <1>       push   esi
5685 00005F77 89CE                    <1>       mov    esi, ecx
5686 00005F79 89C7                    <1>       mov    edi, eax
5687 00005F7B B900040000              <1>       mov    ecx, PAGE_SIZE/4
5688 00005F80 F3A5                    <1>       rep    movsd  ; copy page (4096 bytes)
5689 00005F82 5E                      <1>       pop    esi
5690 00005F83 5F                      <1>       pop    edi
5691                                    <1>       ;
5692                                    <1>
5693                                    <1> ; 17/04/2021
5694                                    <1> ; ('add_to_swap_queue' procedure call is disabled as temporary)
5695                                    <1> ;
5696                                    <1> ;     push   ebx
5697                                    <1> ;     push   eax
5698                                    <1> ;     mov    ebx, ebp
5699                                    <1> ;     ; ebx = virtual address of the memory page
5700                                    <1> ;     call   add_to_swap_queue
5701                                    <1> ;     pop    eax
5702                                    <1> ;     pop    ebx
5703                                    <1>       ;
5704                                    <1>       ;or    ax, PTE_A_USER+PTE_A_PRESENT
5705 00005F84 0C07                    <1>       or     al, PTE_A_USER+PTE_A_WRITE+PTE_A_PRESENT
5706                                    <1> cpt_2:
5707 00005F86 AB                      <1>       stosd  ; EDI points to child's PTE
5708                                    <1>       ;
5709 00005F87 81C500100000            <1>       add    ebp, 4096 ; 20/07/2015 (next page)
5710                                    <1>       ;
5711 00005F8D 39D7                    <1>       cmp    edi, edx
5712 00005F8F 72D2                    <1>       jb     short cpt_0
5713                                    <1> cpt_p_err:
5714 00005F91 59                      <1>       pop    ecx
5715 00005F92 5A                      <1>       pop    edx
5716 00005F93 5F                      <1>       pop    edi
5717 00005F94 5E                      <1>       pop    esi
5718                                    <1>       ;pop   ebx
5719 00005F95 58                      <1>       pop    eax ; *
5720                                    <1> cpt_err:
5721 00005F96 C3                      <1>       retn
5722                                    <1>
5723                                    <1> allocate_memory_block:
5724                                    <1>       ; 01/05/2017
5725                                    <1>       ; 28/04/2017
5726                                    <1>       ; 25/04/2017
5727                                    <1>       ; 01/04/2016, 02/04/2016, 03/04/2016
5728                                    <1>       ; 13/03/2016, 14/03/2016
5729                                    <1>       ; 12/03/2016 (TRDOS 386 = TRDOS v2.0)
5730                                    <1>       ; Allocating contiguous memory pages (in the kernel's memory space)
5731                                    <1>       ;
5732                                    <1>       ; INPUT ->
5733                                    <1>       ;      EAX = Beginning address (physical)
5734                                    <1>       ;      EAX = 0 -> Allocate memory block from the first proper aperture
5735                                    <1>       ;      ECX = Number of bytes to be allocated
5736                                    <1>       ;
5737                                    <1>       ; OUTPUT ->
5738                                    <1>       ;      1) cf = 0 -> successful
5739                                    <1>       ;      EAX = Beginning (physical) address of the allocated memory block
5740                                    <1>       ;      ECX = Number of allocated bytes (rounded up to page borders)
5741                                    <1>       ;      2) cf = 1 -> unsuccessful
5742                                    <1>       ;       2.1) If EAX > 0 ->
5743                                    <1>       ;            (Number of requested pages is more than # of free pages
```

```
                                <1>       ;            but contiguous free pages -the aperture- is not enough!)
                                <1>       ;           EAX = Beginning address of available aperture
                                <1>       ;                (one of all aperture with max. aperture size/length)
                                <1>       ;           ECX = Size of available aperture (memory block) in bytes
                                <1>       ;      2.2) If EAX = 0 -> Out of memory error
                                <1>       ;                (number of free pages is less than requested number)
                                <1>       ;           ECX = Total number of free bytes (free pages * 4096)
                                <1>       ;                (It is not number of contiguous free bytes)
                                <1>       ;
                                <1>       ; (Modified Registers -> EAX, ECX)
                                <1>       ;
                                <1>       ; PURPOSE: Loading a file at memory for copying or running etc.
                                <1>       ; If this procedure returns with cf is set, ECX contains maximum
                                <1>       ; available space and EAX contains the beginning address of it.
                                <1>       ; If EAX has zero, ECX contains total number of free bytes.
                                <1>       ; If requested block has been successfully allocated (by rounding up to
                                <1>       ; the last page border), it must be deallocated later by using
                                <1>       ; 'deallocate_memory_block' procedure.
                                <1>
00005F97 52                     <1>       push   edx ; *
00005F98 BAFF0F0000             <1>       mov    edx, PAGE_SIZE - 1   ; 4095
00005F9D 01D0                   <1>       add    eax, edx
00005F9F 01D1                   <1>       add    ecx, edx
00005FA1 C1E90C                 <1>       shr    ecx, PAGE_SHIFT         ; 12
                                <1>
                                <1>       ; ECX = number of contiguous pages to be allocated
00005FA4 8B15[B0800100]         <1>       mov    edx, [free_pages]
                                <1>       ; 01/05/2017
                                <1>       ;or    ecx, ecx
                                <1>       ;jz    short amb3
                                <1>       ; If ECX=0, set cf to 1 and return with max. available mem block size
                                <1>
00005FAA 39D1                   <1>       cmp    ecx, edx
00005FAC 7760                   <1>       ja     short amb_3
                                <1>
00005FAE C1E80C                 <1>       shr    eax, PAGE_SHIFT      ; 12
                                <1>
00005FB1 89C2                   <1>       mov    edx, eax          ; page number
00005FB3 C1EA03                 <1>       shr    edx, 3            ; to get offset to M.A.T.
                                <1>                                ; (1 allocation bit = 1 page)
                                <1>                                ; (1 allocation bytes = 8 pages)
00005FB6 80E2FC                 <1>       and    dl, 0FCh          ; clear lower 2 bits
                                <1>                                ; (to get 32 bit position)
00005FB9 53                     <1>       push   ebx ; **
                                <1> amb_0:
00005FBA 890D[648C0100]         <1>       mov    [mem_ipg_count], ecx ; initial (reset) value of page count
00005FC0 890D[688C0100]         <1>       mov    [mem_pg_count], ecx
00005FC6 31C9                   <1>       xor    ecx, ecx ; 0
00005FC8 890D[6C8C0100]         <1>       mov    [mem_aperture], ecx ; 0
00005FCE 890D[708C0100]         <1>       mov    [mem_max_aperture], ecx ; 0
                                <1>
00005FD4 BB00001000             <1>       mov    ebx, MEM_ALLOC_TBL   ; Memory Allocation Table address.
00005FD9 3B15[B4800100]         <1>       cmp    edx, [next_page]     ; Is the beginning page address lower
                                <1>                                ; than the address in 'next_page' ?
                                <1>                                ; (the first/next free page of user space)
00005FDF 7208                   <1>       jb     short amb_1
00005FE1 3B15[B8800100]         <1>       cmp    edx, [last_page]     ; is the beginning page address higher
                                <1>                                ; than the address in 'last_page' ?
                                <1>                                ; (end of the memory)
00005FE7 7606                   <1>       jna    short amb_2       ; no
                                <1> amb_1:
00005FE9 8B15[B4800100]         <1>       mov    edx, [next_page]     ; M.A.T. offset (1 M.A.T. byte = 8 pages)
                                <1> amb_2:
00005FEF 01D3                   <1>       add    ebx, edx
                                <1>
                                <1>       ; 28/04/2017
                                <1>       ;xor   ecx, ecx
00005FF1 0FBC0B                 <1>       bsf    ecx, [ebx]        ; 0 to 31
00005FF4 89D0                   <1>       mov    eax, edx
00005FF6 C1E003                 <1>       shl    eax, 3            ; *8
00005FF9 01C8                   <1>       add    eax, ecx          ; beginning page number
                                <1>
00005FFB A3[748C0100]           <1>       mov    [mem_pg_pos], eax    ; beginning page no (for curr. mem. aperture)
00006000 A3[788C0100]           <1>       mov    [mem_max_pg_pos], eax ; beginning page no for max. mem. aperture
                                <1>
00006005 83E01F                 <1>       and    eax, 1Fh          ; lower 5 bits only (0 to 31)
                                <1>                                ; (allocation bit position)
00006008 750E                   <1>       jnz    short amb_4       ; 0
0000600A B120                   <1>       mov    cl, 32
0000600C EB4B                   <1>       jmp    short amb_10
                                <1>
                                <1> amb_3:       ; out_of_memory
0000600E 31C0                   <1>       xor    eax, eax ; 0
00006010 89D1                   <1>       mov    ecx, edx ; free pages
00006012 C1E10C                 <1>       shl    ecx, PAGE_SHIFT
00006015 5A                     <1>       pop    edx ; *
00006016 F9                     <1>       stc
00006017 C3                     <1>       retn
                                <1> amb_4:
00006018 8B13                   <1>       mov    edx, [ebx]
0000601A 88C1                   <1>       mov    cl, al ; 1 to 31
0000601C D3EA                   <1>       shr    edx, cl
0000601E 89D0                   <1>       mov    eax, edx
                                <1> amb_5:
00006020 D1E8                   <1>       shr    eax, 1 ; (***)
00006022 7317                   <1>       jnc    short amb_7
00006024 FF05[6C8C0100]         <1>       inc    dword [mem_aperture]
0000602A FF0D[688C0100]         <1>       dec    dword [mem_pg_count]
00006030 7470                   <1>       jz     short amb_15
                                <1> amb_6:
                                <1>       ; 28/04/2017
00006032 FEC1                   <1>       inc    cl
00006034 80F920                 <1>       cmp    cl, 32
00006037 730D                   <1>       jnb    short amb_9
00006039 EBE5                   <1>       jmp    short amb_5
```

```
5849                                    <1> amb_7:
5850 0000603B 50                        <1>         push   eax ; (***) allocation bits (in shifted status)
5851 0000603C E81B010000                <1>         call   amb_26 ; set maximum memory aperture (free memory block size)
5852 00006041 58                        <1>         pop    eax ; (***)
5853 00006042 EBEE                      <1>         jmp    short amb_6
5854                                    <1> amb_8:
5855                                    <1>         ; 28/04/2017
5856 00006044 B120                      <1>         mov    cl, 32
5857                                    <1> amb_9:
5858 00006046 89DA                      <1>         mov    edx, ebx
5859 00006048 81EA00001000              <1>         sub    edx, MEM_ALLOC_TBL
5860 0000604E 3B15[B8800100]            <1>         cmp    edx, [last_page]
5861 00006054 7336                      <1>         jnb    short amb_14 ; contiguous pages not enough
5862 00006056 83C304                    <1>         add    ebx, 4
5863                                    <1> amb_10:
5864 00006059 8B03                      <1>         mov    eax, [ebx]
5865 0000605B 21C0                      <1>         and    eax, eax
5866 0000605D 7408                      <1>         jz    short amb_11 ; there is not a free page bit in this alloc dword
5867 0000605F 40                        <1>         inc    eax ; 0FFFFFFFFh -> 0
5868 00006060 740C                      <1>         jz     short amb_12 ; all of bits are set (32 free pages)
5869 00006062 48                        <1>         dec    eax
5870 00006063 28C9                      <1>         sub    cl, cl ; 0
5871 00006065 EBB9                      <1>         jmp    short amb_5
5872                                    <1> amb_11:
5873 00006067 E8F0000000                <1>         call   amb_26 ; set maximum memory aperture (free memory block size)
5874 0000606C EBD8                      <1>         jmp    short amb_9
5875                                    <1> amb_12:
5876 0000606E 390D[688C0100]            <1>         cmp    [mem_pg_count], ecx ; 32
5877 00006074 7306                      <1>         jnb    short amb_13
5878 00006076 8B0D[688C0100]            <1>         mov    ecx, [mem_pg_count]
5879                                    <1> amb_13:
5880 0000607C 010D[6C8C0100]            <1>         add    [mem_aperture], ecx
5881 00006082 290D[688C0100]            <1>         sub    [mem_pg_count], ecx
5882 00006088 7618                      <1>         jna    short amb_15
5883 0000608A EBBA                      <1>         jmp    short amb_9 ; 01/05/2017
5884                                    <1> amb_14:
5885 0000608C E8CB000000                <1>         call   amb_26 ; 28/04/2017
5886 00006091 A1[788C0100]              <1>         mov    eax, [mem_max_pg_pos] ; begin address of max. mem aperture
5887 00006096 8B0D[708C0100]            <1>         mov    ecx, [mem_max_aperture] ; max. (largest) memory aperture
5888 0000609C F9                        <1>         stc
5889 0000609D E9AF000000                <1>         jmp   amb_25
5890                                    <1>
5891                                    <1> amb_15: ; OK !
5892 000060A2 A1[748C0100]              <1>         mov    eax, [mem_pg_pos]    ; Beginning address as page number
5893 000060A7 8B0D[6C8C0100]            <1>         mov    ecx, [mem_aperture]  ; Free contiguous page count (>=1)
5894                                    <1> amb_16:
5895                                    <1>         ; allocate contiguous memory pages (via memory allocation table bits)
5896 000060AD 89C2                      <1>         mov    edx, eax
5897                                    <1>         ; 25/04/2017
5898 000060AF C1EA03                    <1>         shr    edx, 3        ; 8 pages in one allocation byte
5899 000060B2 80E2FC                    <1>         and    dl, 0FCh      ; clear lower 2 bits
5900                                    <1>                              ; (for dword/32bit positioning)
5901                                    <1>
5902 000060B5 BB00001000                <1>         mov    ebx, MEM_ALLOC_TBL
5903 000060BA 01D3                      <1>         add    ebx, edx
5904 000060BC 83E01F                    <1>         and    eax, 1Fh ; 31
5905                                    <1>         ; 03/04/2016
5906 000060BF BA20000000                <1>         mov    edx, 32
5907 000060C4 28C2                      <1>         sub    dl, al
5908 000060C6 39CA                      <1>         cmp    edx, ecx      ; ecx >= 1
5909 000060C8 7602                      <1>         jna    short amb_17
5910 000060CA 89CA                      <1>         mov    edx, ecx
5911                                    <1> amb_17:
5912 000060CC 29D1                      <1>         sub    ecx, edx
5913 000060CE 51                        <1>         push   ecx ; ***
5914 000060CF 89D1                      <1>         mov    ecx, edx
5915                                    <1> amb_18:
5916 000060D1 0FB303                    <1>         btr    [ebx], eax    ; The destination bit indexed by the source value
5917                                    <1>                              ; is copied into the Carry Flag and then cleared
5918                                    <1>                              ; in the destination.
5919 000060D4 FF0D[B0800100]            <1>         dec    dword [free_pages] ; 1 page has been allocated (X = X-1)
5920 000060DA 49                        <1>         dec    ecx
5921 000060DB 7404                      <1>         jz     short amb_19
5922 000060DD FEC0                      <1>         inc    al
5923 000060DF EBF0                      <1>         jmp    short amb_18
5924                                    <1> amb_19:
5925 000060E1 59                        <1>         pop    ecx ; ***
5926 000060E2 21C9                      <1>         and    ecx, ecx ; 0 ?
5927 000060E4 741E                      <1>         jz     short amb_22
5928                                    <1>         ; 01/04/2016
5929 000060E6 B020                      <1>         mov    al, 32
5930                                    <1> amb_20:
5931 000060E8 83C304                    <1>         add    ebx, 4
5932 000060EB 39C1                      <1>         cmp    ecx, eax ; 32
5933 000060ED 7305                      <1>         jnb    short amb_21
5934                                    <1>         ; ECX < 32
5935 000060EF 28C0                      <1>         sub    al, al ; 0
5936 000060F1 50                        <1>         push   eax ; 0 ***
5937 000060F2 EBDD                      <1>         jmp    short amb_18
5938                                    <1> amb_21:
5939 000060F4 2905[B0800100]            <1>         sub    [free_pages], eax   ; [free_pages] = [free_pages] - 32
5940 000060FA C70300000000              <1>         mov    dword [ebx], 0          ; reset 32 bits
5941 00006100 29C1                      <1>         sub    ecx, eax ; 32
5942 00006102 75E4                      <1>         jnz    short amb_20
5943                                    <1> amb_22:
5944 00006104 A1[748C0100]              <1>         mov    eax, [mem_pg_pos]   ; Beginning address as page number
5945 00006109 8B0D[6C8C0100]            <1>         mov    ecx, [mem_aperture] ; Free contiguous page count
5946                                    <1>         ; [next_page] update
5947 0000610F 89C2                      <1>         mov    edx, eax
5948                                    <1>         ; 03/04/2016
5949 00006111 C1EA03                    <1>         shr    edx, 3            ; to get offset to M.A.T.
5950                                    <1>                                  ; (1 allocation bit = 1 page)
5951                                    <1>                                  ; (1 allocation bytes = 8 pages)
5952 00006114 80E2FC                    <1>         and    dl, 0FCh          ; clear lower 2 bits
5953                                    <1>                                  ; (to get 32 bit position)
```

```
5954 00006117 3B15[B4800100]      <1>       cmp    edx, [next_page] ; first free page pointer offset
5955 0000611D 7732                <1>       ja     short amb_25
5956 0000611F BB00001000          <1>       mov    ebx, MEM_ALLOC_TBL
5957 00006124 833C1300            <1>       cmp    dword [ebx+edx], 0
5958 00006128 7721                <1>       ja     short amb_24
5959 0000612A 89C2                <1>       mov    edx, eax
5960 0000612C 01CA                <1>       add    edx, ecx
5961 0000612E C1EA03              <1>       shr    edx, 3
5962 00006131 80E2FC              <1>       and    dl, 0FCh
5963                               <1> amb_23:
5964 00006134 833C1300            <1>       cmp    dword [ebx+edx], 0
5965 00006138 7711                <1>       ja     short amb_24
5966 0000613A 83C204              <1>       add    edx, 4
5967 0000613D 3B15[B8800100]      <1>       cmp    edx, [last_page]    ; last page pointer offset
5968 00006143 76EF                <1>       jna    short amb_23
5969 00006145 8B15[BC800100]      <1>       mov    edx, [first_page]   ; (for) beginning of user's space
5970                               <1> amb_24:
5971 0000614B 8915[B4800100]      <1>       mov    [next_page], edx
5972                               <1> amb_25:
5973 00006151 9C                  <1>       pushf
5974 00006152 C1E00C              <1>       shl    eax, PAGE_SHIFT          ; convert to phy. address in bytes
5975 00006155 C1E10C              <1>       shl    ecx, PAGE_SHIFT          ; convert to byte counts
5976 00006158 9D                  <1>       popf
5977 00006159 5B                  <1>       pop    ebx ; **
5978 0000615A 5A                  <1>       pop    edx ; *
5979 0000615B C3                  <1>       retn
5980                               <1>
5981                               <1> amb_26:     ; set maximum free memory aperture (free memory block size)
5982 0000615C 89DA                <1>       mov    edx, ebx ; current address
5983 0000615E 81EA00001000        <1>       sub    edx, MEM_ALLOC_TBL ; MAT beginning address
5984                               <1>       ; 02/04/2016
5985 00006164 C1E203              <1>       shl    edx, 3 ; MAT byte offset * 8 = page number base
5986 00006167 01CA                <1>       add    edx, ecx ; current page number (ecx = 0 to 32)
5987                               <1>       ;
5988 00006169 A1[6C8C0100]        <1>       mov    eax, [mem_aperture]
5989 0000616E 21C0                <1>       and    eax, eax
5990 00006170 7421                <1>       jz    short amb_27
5991 00006172 C705[6C8C0100]0000- <1>       mov   dword [mem_aperture], 0
5991 0000617A 0000                <1>
5992 0000617C 3B05[708C0100]      <1>       cmp    eax, [mem_max_aperture]
5993 00006182 760F                <1>       jna    short amb_27
5994 00006184 A3[708C0100]        <1>       mov    [mem_max_aperture], eax
5995                               <1>       ; 25/04/2017
5996 00006189 A1[748C0100]        <1>       mov    eax, [mem_pg_pos]
5997                               <1>       ; EAX = Beginning page number of the max. aperture
5998 0000618E A3[788C0100]        <1>       mov    [mem_max_pg_pos], eax
5999                               <1> amb_27:
6000 00006193 8915[748C0100]      <1>       mov    [mem_pg_pos], edx ; current page
6001                               <1>
6002 00006199 A1[648C0100]        <1>       mov    eax, [mem_ipg_count] ; initial (reset) value of page count
6003 0000619E A3[688C0100]        <1>       mov    [mem_pg_count], eax
6004                               <1>
6005 000061A3 C3                  <1>       retn
6006                               <1>
6007                               <1> deallocate_memory_block:
6008                               <1>       ; 03/04/2016
6009                               <1>       ; 14/03/2016 (TRDOS 386 = TRDOS v2.0)
6010                               <1>       ; Deallocating contiguous memory pages (in the kernel's memory space)
6011                               <1>       ;
6012                               <1>       ; INPUT ->
6013                               <1>       ;     EAX = Beginning address (physical)
6014                               <1>       ;     ECX = Number of bytes to be deallocated
6015                               <1>       ;
6016                               <1>       ; OUTPUT ->
6017                               <1>       ;     Memory Allocation Table bits will be updated
6018                               <1>       ;     [free_pages] will be changed (increased)
6019                               <1>       ;
6020                               <1>       ; (Modified Registers -> EAX, ECX)
6021                               <1>       ;
6022                               <1>       ; PURPOSE: Unloading/Freeing a file -or an allocated memory block-
6023                               <1>       ; at memory after copying, running, saving, reading, writing etc.
6024                               <1>       ;
6025                               <1>
6026 000061A4 52                  <1>       push   edx ; *
6027 000061A5 53                  <1>       push   ebx ; **
6028                               <1>
6029 000061A6 C1E80C              <1>       shr    eax, PAGE_SHIFT          ; 12
6030 000061A9 C1E90C              <1>       shr    ecx, PAGE_SHIFT          ; 12
6031                               <1>
6032                               <1>       ; EAX = Beginning page number
6033                               <1>       ; ECX = Number of contiguous pages to be deallocated
6034                               <1> damb_0:
6035                               <1>       ; deallocate contiguous memory pages (via memory allocation table bits)
6036 000061AC 89C2                <1>       mov    edx, eax
6037 000061AE C1EA03              <1>       shr    edx, 3            ; to get offset to M.A.T.
6038                               <1>                                ; (1 allocation bit = 1 page)
6039                               <1>                                ; (1 allocation bytes = 8 pages)
6040 000061B1 80E2FC              <1>       and    dl, 0FCh          ; clear lower 2 bits
6041                               <1>                                ; (to get 32 bit position)
6042 000061B4 3B15[B4800100]      <1>       cmp    edx, [next_page] ; next free page
6043 000061BA 7306                <1>       jnb    short damb_1
6044 000061BC 8915[B4800100]      <1>       mov    [next_page], edx
6045                               <1> damb_1:
6046 000061C2 BB00001000          <1>       mov    ebx, MEM_ALLOC_TBL
6047 000061C7 01D3                <1>       add    ebx, edx
6048 000061C9 83E01F              <1>       and    eax, 1Fh ; 31
6049                               <1>
6050                               <1>       ; 03/04/2016
6051 000061CC BA20000000          <1>       mov    edx, 32
6052 000061D1 28C2                <1>       sub    dl, al
6053 000061D3 39CA                <1>       cmp    edx, ecx
6054 000061D5 7602                <1>       jna    short damb_2
6055 000061D7 89CA                <1>       mov    edx, ecx
6056                               <1> damb_2:
6057 000061D9 29D1                <1>       sub    ecx, edx
```

```
6058 000061DB 51                  <1>       push   ecx ; ***
6059 000061DC 89D1                <1>       mov    ecx, edx
6060                              <1> damb_3:
6061 000061DE 0FAB03              <1>       bts    [ebx], eax        ; unlink/release/deallocate page
6062                              <1>                                ; set relevant bit to 1.
6063                              <1>                                ; set CF to the previous bit value
6064 000061E1 FF05[B0800100]      <1>       inc    dword [free_pages]   ; 1 page has been deallocated (X = X+1)
6065 000061E7 49                  <1>       dec    ecx
6066 000061E8 7404                <1>       jz     short damb_4
6067 000061EA FEC0                <1>       inc    al
6068 000061EC EBF0                <1>       jmp    short damb_3
6069                              <1> damb_4:
6070 000061EE 59                  <1>       pop    ecx ; ***
6071 000061EF 21C9                <1>       and    ecx, ecx ; 0 ?
6072 000061F1 741E                <1>       jz     short damb_7
6073                              <1>       ; 03/04/2016
6074 000061F3 B020                <1>       mov    al, 32
6075                              <1> damb_5:
6076 000061F5 83C304              <1>       add    ebx, 4
6077 000061F8 39C1                <1>       cmp    ecx, eax ; 32
6078 000061FA 7305                <1>       jnb    short damb_6
6079                              <1>       ; ECX < 32
6080 000061FC 28C0                <1>       sub    al, al ; 0
6081 000061FE 50                  <1>       push   eax ; 0 ***
6082 000061FF EBDD                <1>       jmp    short damb_3
6083                              <1> damb_6:
6084 00006201 0105[B0800100]      <1>       add    [free_pages], eax ; [free_pages] = [free_pages] + 32
6085 00006207 C703FFFFFFFF        <1>       mov    dword [ebx], 0FFFFFFFFh ; set 32 bits
6086 0000620D 29C1                <1>       sub    ecx, eax ; 32
6087 0000620F 75E4                <1>       jnz    short damb_5
6088                              <1> damb_7:
6089 00006211 5B                  <1>       pop    ebx ; **
6090 00006212 5A                  <1>       pop    edx ; *
6091 00006213 C3                  <1>       retn
6092                              <1>
6093                              <1> direct_memory_access:
6094                              <1>       ; 17/04/2021 (temporary modifications)
6095                              <1>       ; 22/07/2017
6096                              <1>       ; 12/05/2017
6097                              <1>       ; 16/07/2016
6098                              <1>       ; 12/07/2016 (TRDOS 386 = TRDOS v2.0)
6099                              <1>       ; This processure will be called to map
6100                              <1>       ; user's (ring 3) page tables to access phsical
6101                              <1>       ; (flat/linear) memory addresses, directly (without
6102                              <1>       ;  kernel's data transfer functions).
6103                              <1>       ;
6104                              <1>       ; Purpose: Video memory access and shared memory access.
6105                              <1>       ;
6106                              <1>       ; INPUT ->
6107                              <1>       ;     EAX = Beginning address (physical).
6108                              <1>       ;     EBX = User's buffer address ; 12/05/2017
6109                              <1>       ;     ECX = Number of contiguous pages to be mapped.
6110                              <1>       ; OUTPUT ->
6111                              <1>       ;     User's page directory and pages tables
6112                              <1>       ;     will be updated.
6113                              <1>       ;
6114                              <1>       ;     If an old page table entry has valid page address,
6115                              <1>       ;     that page will be deallocated just before PTE will
6116                              <1>       ;     be changed for direct (1 to 1) memory page access.
6117                              <1>       ;
6118                              <1>       ;     If old PTE value points to a swapped page,
6119                              <1>       ;       that page (block) will be unlinked on swap disk.
6120                              <1>       ;
6121                              <1>       ;     Newly allocated pages (except page tables) will not
6122                              <1>       ;     be applied to Memory Allocation Table.
6123                              <1>       ;     AVL bit 1 (PTE bit 10) of page table entry will be
6124                              <1>       ;     used to indicate shared (direct) memory page; then,
6125                              <1>       ;     this page will not be deallocated later during
6126                              <1>       ;     process termination. (Memory Allocation Table and
6127                              <1>       ;     free memory count will not be affected.
6128                              <1>       ;     (Except deallocating page table's itself.)
6129                              <1>       ;
6130                              <1>       ;     CF = 1 -> error (EAX = error code)
6131                              <1>       ;     CF = 0 -> success (EAX = beginning address)
6132                              <1>       ;
6133                              <1>       ;; (Modified Registers -> none)
6134                              <1>       ; Modified registers: ebp, edx, ecx, ebx, esi, edi
6135                              <1>       ;
6136                              <1>
6137                              <1>       ;push  ebp
6138                              <1>       ;push  ebx
6139                              <1>       ;push  ecx
6140                              <1>       ;push  edx
6141 00006214 662500F0            <1>       and    ax, PTE_A_CLEAR ; clear page offset
6142 00006218 50                  <1>       push   eax
6143                              <1>       ;and   ecx, ecx ; page count
6144                              <1>       ;jz    dmem_acc_7  ; 'insufficient memory' error
6145 00006219 89C5                <1>       mov    ebp, eax
6146 0000621B 81C300004000        <1>       add    ebx, CORE ; 12/05/2017
6147                              <1> dmem_acc_0:
6148 00006221 891D[14940100]      <1>       mov    [base_addr], ebx ; 12/05/2017
6149 00006227 A1[B8030300]        <1>       mov    eax, [u.pgdir] ; page dir address (physical)
6150 0000622C E856F9FFFF          <1>       call   get_pte
6151                              <1>              ; EDX = Page table entry address (if CF=0)
6152                              <1>              ;       Page directory entry address (if CF=1)
6153                              <1>              ;       (Bit 0 value is 0 if PT is not present)
6154                              <1>              ; EAX = Page table entry value (page address)
6155                              <1>              ;     CF = 1 -> PDE not present or invalid ?
6156 00006231 7321                <1>       jnc    short dmem_acc_1
6157                              <1>       ;
6158 00006233 E83DF8FFFF          <1>       call   allocate_page
6159                              <1>       ;jc    dmem_acc_7  ; 'insufficient memory' error
6160                              <1>       ; 17/04/2021
6161 00006238 7215                <1>       jc     short _dmem_acc_7
6162                              <1>       ;
```

```
6163 0000623A E8A7F8FFFF          <1>       call   clear_page
6164                              <1>       ; EAX = Physical (base) address of the allocated (new) page
6165 0000623F 0C07                <1>       or     al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER ; 4+2+1 = 7
6166                              <1>                        ; lower 3 bits are used as U/S, R/W, P flags
6167                              <1>                        ; (user, writable, present page)
6168 00006241 8902                <1>       mov    [edx], eax ; Let's put the new page directory entry here !
6169 00006243 A1[B8030300]        <1>       mov    eax, [u.pgdir]
6170 00006248 E83AF9FFFF          <1>       call   get_pte
6171                              <1>       ;jc    dmem_acc_7 ; 'insufficient memory' error
6172                              <1>       ; 17/04/2021
6173 0000624D 7305                <1>       jnc    short dmem_acc_1
6174                              <1> _dmem_acc_7:
6175 0000624F E985000000          <1>       jmp    dmem_acc_7
6176                              <1> dmem_acc_1:
6177                              <1>       ; EAX = PTE value, EDX = PTE address
6178 00006254 A801                <1>       test   al, PTE_A_PRESENT
6179                              <1>       ;jnz   short dmem_acc_2   ; 17/04/2021 (*)
6180                              <1>       ; 17/04/2021 (temporary)
6181 00006256 745F                <1>       jz     short short dmem_acc_6  ; ! temporary ! (*)
6182                              <1>
6183                              <1> ; 17/04/2021
6184                              <1> ; (following code is disabled as temporary)
6185                              <1> ;
6186                              <1> ;     or     eax, eax
6187                              <1> ;     jz     short dmem_acc_6   ; Change PTE
6188                              <1> ;     shr    eax, 1       ; swap disk block (8 sectors) address
6189                              <1> ;     ; unlink swap disk block
6190                              <1> ;     call   unlink_swap_block
6191                              <1> ;     jmp    short dmem_acc_6
6192                              <1>
6193                              <1> dmem_acc_2:
6194 00006258 A802                <1>       test   al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
6195                              <1>                              ; (must be 1)
6196 0000625A 7550                <1>       jnz    short dmem_acc_4
6197                              <1>       ; Read only -duplicated- page (belongs to a parent or a child)
6198 0000625C 66A90002            <1>       test  ax, PTE_DUPLICATED ; Was this page duplicated
6199                              <1>                                ; as child's page ?
6200 00006260 7455                <1>       jz     short dmem_acc_5 ; Change PTE but don't deallocate the page!
6201                              <1>
6202                              <1>       ;push  edi
6203                              <1>       ;push  esi
6204                              <1>
6205 00006262 51                  <1>       push   ecx
6206                              <1>       ;push  ebx
6207 00006263 8B1D[BC030300]      <1>       mov    ebx, [u.ppgdir] ; parent's page dir address (physical)
6208                              <1>
6209                              <1>       ; check the parent's PTE value is read only & same page or not..
6210 00006269 89EF                <1>       mov    edi, ebp
6211 0000626B C1EF16              <1>       shr    edi, PAGE_D_SHIFT ; 22
6212                              <1>       ; EDI = page directory entry index (0-1023)
6213 0000626E 89EE                <1>       mov    esi, ebp
6214 00006270 C1EE0C              <1>       shr    esi, PAGE_SHIFT ; 12
6215 00006273 81E6FF030000        <1>       and    esi, PTE_MASK
6216                              <1>       ; ESI = page table entry index (0-1023)
6217                              <1>
6218 00006279 66C1E702            <1>       shl    di, 2 ; * 4
6219 0000627D 01FB                <1>       add    ebx, edi ; PDE offset (for the parent)
6220 0000627F 8B0F                <1>       mov    ecx, [edi]
6221 00006281 F6C101              <1>       test   cl, PDE_A_PRESENT ; present (valid) or not ?
6222 00006284 7425                <1>       jz     short dmem_acc_3    ; parent process does not use this page
6223 00006286 6681E100F0          <1>       and    cx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
6224 0000628B 66C1E602            <1>       shl    si, 2 ; *4
6225 0000628F 01CE                <1>       add    esi, ecx ; PTE offset (for the parent)
6226 00006291 8B1E                <1>       mov    ebx, [esi]
6227 00006293 F6C301              <1>       test   bl, PTE_A_PRESENT ; present or not ?
6228 00006296 7413                <1>       jz     short dmem_acc_3    ; parent process does not use this page
6229 00006298 662500F0            <1>       and    ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
6230 0000629C 6681E300F0          <1>       and    bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
6231 000062A1 39D8                <1>       cmp    eax, ebx     ; parent's and child's pages are same ?
6232 000062A3 7506                <1>       jne    short dmem_acc_3    ; not same page
6233                              <1>                                ; deallocate the child's page
6234 000062A5 800E02              <1>       or    byte [esi], PTE_A_WRITE ; convert to writable page (parent)
6235                              <1>       ;pop   ebx
6236 000062A8 59                  <1>       pop    ecx
6237 000062A9 EB0C                <1>       jmp    short dmem_acc_5
6238                              <1> dmem_acc_3:
6239                              <1>       ;pop   ebx
6240 000062AB 59                  <1>       pop    ecx
6241                              <1> dmem_acc_4:
6242 000062AC 66A90004            <1>       test   ax, PTE_SHARED ; shared or direct memory access indicator
6243 000062B0 7505                <1>       jnz    short dmem_acc_5   ; AVL bit 1 = 1, do not deallocate this page!
6244                              <1>       ;
6245                              <1>       ;and   ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
6246 000062B2 E886F9FFFF          <1>       call   deallocate_page
6247                              <1> dmem_acc_5:
6248                              <1>       ;pop   esi
6249                              <1>       ;pop   edi
6250                              <1> dmem_acc_6:
6251 000062B7 89E8                <1>       mov    eax, ebp ; physical page (offset=0) address
6252                              <1>       ; EAX = memory page address
6253                              <1>       ; EDX = PTE entry address (physical)
6254 000062B9 660D0704            <1>       or     ax, PTE_A_PRESENT+PTE_A_USER+PTE_A_WRITE+PTE_SHARED
6255                              <1>                     ; present flag, bit 0 = 1
6256                              <1>                     ; user flag, bit 2 = 1
6257                              <1>                     ; writable flag, bit 1 = 1
6258                              <1>                     ; direct memory access flag, bit 10 = 1
6259                              <1>                     ; (This page must not be deallocated!)
6260 000062BD 8902                <1>       mov    [edx], eax  ; Update PTE value
6261 000062BF 49                  <1>       dec    ecx ; remain count of contiguous pages
6262 000062C0 741E                <1>       jz     short dmem_acc_8
6263 000062C2 81C500100000        <1>       add    ebp, PAGE_SIZE ; next physical page address
6264                              <1>       ; 22/07/2017
6265                              <1>       ;mov   eax, ebp
6266                              <1>       ; 12/05/2017
6267 000062C8 8B1D[14940100]      <1>       mov    ebx, [base_addr] ; linear address (virtual+CORE)
```

```
6268 000062CE 81C300100000        <1>       add    ebx, PAGE_SIZE      ; next linear address
6269 000062D4 E948FFFFFF          <1>       jmp    dmem_acc_0
6270                              <1> dmem_acc_7:  ; ERROR !
6271 000062D9 C7042404000000      <1>       mov    dword [esp], ERR_MINOR_IM
6272                              <1>              ; Insufficient memory (minor) error!
6273                              <1>              ; Major error = 0 (No protection fault)
6274                              <1>       ; cf = 1
6275                              <1> dmem_acc_8:
6276 000062E0 58                  <1>       pop    eax
6277                              <1>       ;pop   edx
6278                              <1>       ;pop   ecx
6279                              <1>       ;pop   ebx
6280                              <1>       ;pop   ebp
6281 000062E1 C3                  <1>       retn
6282                              <1>
6283                              <1> deallocate_user_pages:
6284                              <1>       ; 20/05/2017
6285                              <1>       ; 15/05/2017
6286                              <1>       ; 20/02/2017
6287                              <1>       ; 19/02/2017 (TRDOS 386 = TRDOS v2.0)
6288                              <1>       ;
6289                              <1>       ; Deallocate virtually contiguous user pages (memory block)
6290                              <1>       ; (caller: 'sysdalloc' system call)
6291                              <1>       ;
6292                              <1>       ; INPUT ->
6293                              <1>       ;      EBX = VIRTUAL ADDRESS (beginning address)
6294                              <1>       ;      ECX = byte count
6295                              <1>       ;      [u.pgdir] = user's page directory
6296                              <1>       ;      [u.ppdir] = parent's page directory
6297                              <1>       ;
6298                              <1>       ; OUTPUT ->
6299                              <1>       ;    If CF = 0
6300                              <1>       ;      EAX = Deallocated memory bytes
6301                              <1>       ;        (Even if shared or read only pages will not be
6302                              <1>       ;         deallocated on M.A.T., this byte count will be
6303                              <1>       ;         returned as virtually deallocated bytes; in fact
6304                              <1>       ;         virtually deallocated user pages * 4096.)
6305                              <1>       ;      EBX = Virtual address (as rounded up)
6306                              <1>       ;    If CF = 1
6307                              <1>       ;      EAX = 0 (there is not any deallocated pages)
6308                              <1>       ;
6309                              <1>       ; Note: Empty page tables will not be deallocated!!!
6310                              <1>       ;     (they will be deallocated at process termination stage)
6311                              <1>       ;
6312                              <1>       ; Modified Registers -> EAX, EDX, ESI, EDI, EBX, ECX, EBP
6313                              <1>       ;
6314 000062E2 89DE                <1>       mov    esi, ebx
6315 000062E4 89F7                <1>       mov    edi, esi
6316 000062E6 01CF                <1>       add    edi, ecx
6317 000062E8 81C6FF0F0000        <1>       add    esi, PAGE_SIZE - 1  ; 4095 (round up)
6318 000062EE C1EE0C              <1>       shr    esi, PAGE_SHIFT
6319 000062F1 C1EF0C              <1>       shr    edi, PAGE_SHIFT
6320 000062F4 89F8                <1>       mov    eax, edi ; end page
6321 000062F6 29F0                <1>       sub    eax, esi ; end page - start page
6322 000062F8 0F86C8000000        <1>       jna    da_u_pd_err  ; < 1
6323 000062FE 89F3                <1>       mov    ebx, esi
6324 00006300 C1E30C              <1>       shl    ebx, PAGE_SHIFT ; virtual address (as rounded up)
6325 00006303 53                  <1>       push   ebx ; *
6326 00006304 89C1                <1>       mov    ecx, eax ; page count
6327 00006306 C1E00C              <1>       shl    eax, PAGE_SHIFT ; byte count as adjusted
6328 00006309 50                  <1>       push   eax ; **
6329 0000630A 8B1D[B8030300]      <1>       mov    ebx, [u.pgdir] ; physical addr of user's page dir
6330 00006310 81C600040000        <1>       add    esi, CORE/PAGE_SIZE
6331 00006316 89F7                <1>       mov    edi, esi
6332 00006318 81E7FF030000        <1>       and    edi, PTE_MASK ; PTE entry in the page table
6333 0000631E 57                  <1>       push   edi ; *** ; PTE index (of page directory)
6334 0000631F C1EE0A              <1>       shr    esi, PAGE_D_SHIFT - PAGE_SHIFT ; 22-12=10
6335 00006322 89F2                <1>       mov    edx, esi
6336                              <1>       ; EDX = PDE index
6337 00006324 C1E602              <1>       shl    esi, 2 ; convert PDE index to dword offset
6338 00006327 01DE                <1>       add    esi, ebx ; add page directory address
6339                              <1> da_u_pd_1:
6340 00006329 AD                  <1>       lodsd
6341                              <1>       ;
6342 0000632A 89F5                <1>       mov    ebp, esi ; 20/02/2017
6343                              <1>       ; EBP = next PDE address
6344                              <1>       ;
6345 0000632C A801                <1>       test   al, PDE_A_PRESENT ; bit 0, present flag (must be 1)
6346 0000632E 0F8487000000        <1>       jz     da_u_pd_3 ; 20/05/2017
6347 00006334 662500F0            <1>       and    ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
6348                              <1>       ; EAX = PHYSICAL (flat) ADDRESS OF THE PAGE TABLE
6349 00006338 8B3C24              <1>       mov    edi, [esp] ; ***
6350                              <1>       ; EDI = PTE index (of complete page directory)
6351                              <1>       ;and   edi, PTE_MASK ; PTE entry in the page table
6352 0000633B C1E702              <1>       shl    edi, 2 ; convert PTE index to dword offset
6353 0000633E 89FE                <1>       mov    esi, edi ; PTE offset in page table (0-4092)
6354 00006340 01C6                <1>       add    esi, eax ; now, esi points to requested PTE
6355                              <1> da_u_pt_0:
6356 00006342 AD                  <1>       lodsd
6357 00006343 A801                <1>       test   al, PTE_A_PRESENT ; bit 0, present flag (must be 1)
6358 00006345 7452                <1>       jz     short da_u_pt_1
6359                              <1>       ;
6360 00006347 A802                <1>       test   al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
6361                              <1>                                ; (must be 1)
6362 00006349 753C                <1>       jnz    short da_u_pt_3
6363                              <1>       ; Read only -duplicated- page (belongs to a parent or a child)
6364 0000634B 66A90002            <1>       test  ax, PTE_DUPLICATED ; Was this page duplicated
6365                              <1>                                ; as child's page ?
6366 0000634F 7441                <1>       jz     short da_u_pt_4 ; Clear PTE but don't deallocate the page!
6367                              <1>       ;
6368                              <1>       ; check the parent's PTE value is read only & same page or not..
6369                              <1>       ; EDX = page directory entry index (0-1023)
6370 00006351 52                  <1>       push   edx ; ****
6371                              <1>       ; EDI = page table entry offset (0-4092)
6372 00006352 8B1D[BC030300]      <1>       mov    ebx, [u.ppgdir] ; page directory of the parent process
```

```
6373 00006358 66C1E202            <1>       shl    dx, 2 ; *4
6374 0000635C 01D3                <1>       add    ebx, edx ; PDE address (for the parent)
6375 0000635E 8B13                <1>       mov    edx, [ebx] ; page table address
6376 00006360 F6C201              <1>       test   dl, PDE_A_PRESENT ; present (valid) or not ?
6377 00006363 7421                <1>       jz     short da_u_pt_2     ; parent process does not use this page
6378 00006365 6681E200F0          <1>       and    dx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
6379                              <1>       ; EDI = page table entry offset (0-4092)
6380 0000636A 01D7                <1>       add    edi, edx     ; PTE address (for the parent)
6381 0000636C 8B1F                <1>       mov    ebx, [edi]
6382 0000636E F6C301              <1>       test   bl, PTE_A_PRESENT ; present or not ?
6383 00006371 7413                <1>       jz     short da_u_pt_2     ; parent process does not use this page
6384 00006373 662500F0            <1>       and    ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
6385 00006377 6681E300F0          <1>       and    bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
6386 0000637C 39D8                <1>       cmp    eax, ebx     ; parent's and child's pages are same ?
6387 0000637E 7506                <1>       jne    short da_u_pt_2     ; not same page
6388                              <1>                              ; deallocate the child's page
6389 00006380 800F02              <1>       or    byte [edi], PTE_A_WRITE ; convert to writable page (parent)
6390 00006383 5A                  <1>       pop    edx ; ****
6391 00006384 EB0C                <1>       jmp    short da_u_pt_4
6392                              <1>
6393                              <1> ; 17/04/2021
6394                              <1> ; ('da_u_pt_1' is disabled as temporary)
6395                              <1>
6396                              <1> ;da_u_pt_1:
6397                              <1> ;     or     eax, eax     ; swapped page ?
6398                              <1> ;     jz     short da_u_pt_5     ; no
6399                              <1> ;                              ; yes
6400                              <1> ;     shr    eax, 1
6401                              <1> ;     call   unlink_swap_block ; Deallocate swapped page block
6402                              <1> ;                                ; on the swap disk (or in file)
6403                              <1> ;     jmp    short da_u_pt_5
6404                              <1> da_u_pt_2:
6405 00006386 5A                  <1>       pop    edx ; ****
6406                              <1> da_u_pt_3:
6407 00006387 66A90004            <1>       test   ax, PTE_SHARED      ; shared or direct memory access indicator
6408 0000638B 7505                <1>       jnz    short da_u_pt_4     ; AVL bit 1 = 1, do not deallocate this page!
6409                              <1>       ;
6410                              <1>       ;and   ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
6411 0000638D E8ABF8FFFF          <1>       call   deallocate_page ; set the mem allocation bit of this page
6412                              <1> da_u_pt_4:
6413 00006392 C746FC00000000      <1>       mov    dword [esi-4], 0 ; clear/reset PTE (child, dupl. as parent)
6414                              <1> ; 17/04/2021 (temporary)
6415                              <1> da_u_pt_1:
6416                              <1> da_u_pt_5:
6417                              <1>       ; 20/05/2017
6418 00006399 58                  <1>       pop    eax ; *** PTE index (of page directory)
6419 0000639A 49                  <1>       dec    ecx ; remain page count
6420 0000639B 7426                <1>       jz     short da_u_pd_4
6421 0000639D 40                  <1>       inc    eax ; next PTE
6422 0000639E 6625FF03            <1>       and    ax, PTE_MASK ; PTE entry index in the page table
6423 000063A2 50                  <1>       push   eax ; *** (save again)
6424                              <1>       ;mov   edi, eax
6425                              <1>       ;and   di, PTE_MASK
6426                              <1>       ;cmp   edi, PAGE_SIZE / 4 ; 1024
6427                              <1>       ;jnb   short da_u_pd_2
6428 000063A3 89C7                <1>       mov    edi, eax
6429 000063A5 C1E702              <1>       shl    edi, 2 ; convert index to dword offset
6430                              <1>       ;test  ax, PTE_MASK ; 3FFh
6431 000063A8 09C0                <1>       or     eax, eax
6432 000063AA 7596                <1>       jnz    short da_u_pt_0 ; 1-1023
6433                              <1> da_u_pd_2:
6434 000063AC 42                  <1>       inc    edx
6435                              <1>       ; 20/05/2017
6436 000063AD 6681E2FF03          <1>       and    dx, PTE_MASK  ; 3FFh
6437 000063B2 740F                <1>       jz     short da_u_pd_4  ; 0 (1024)
6438                              <1>       ;cmp   edx, 1024
6439                              <1>       ;jnb   short da_u_pd_4
6440 000063B4 89EE                <1>       mov    esi, ebp ; 20/02/2017
6441 000063B6 E96EFFFFFF          <1>       jmp    da_u_pd_1
6442                              <1> da_u_pd_3:
6443                              <1>       ; 15/05/2017 (empty page directory entry)
6444 000063BB 81E900040000        <1>       sub    ecx, 1024
6445 000063C1 77E9                <1>       ja     short da_u_pd_2 ; 20/05/2017
6446                              <1> da_u_pd_4:
6447 000063C3 58                  <1>       pop    eax ; **
6448 000063C4 5B                  <1>       pop    ebx ; *
6449 000063C5 C3                  <1>       retn
6450                              <1>
6451                              <1> da_u_pd_err:
6452 000063C6 31C0                <1>       xor    eax, eax
6453 000063C8 F9                  <1>       stc
6454 000063C9 C3                  <1>       retn
6455                              <1>
6456                              <1> allocate_user_pages:
6457                              <1>       ; 20/05/2017
6458                              <1>       ; 01/05/2017, 02/05/2017, 15/05/2017
6459                              <1>       ; 04/03/2017
6460                              <1>       ; 20/02/2017 (TRDOS 386 = TRDOS v2.0)
6461                              <1>       ;
6462                              <1>       ; Allocate physically contiguous user pages (memory block)
6463                              <1>       ; (caller: 'sysalloc' system call)
6464                              <1>       ;
6465                              <1>       ; Note: This procedure does not alloc a page's itself
6466                              <1>       ;     (page bit) on Memory Allocation Table.
6467                              <1>       ;     (allocate_memory_block is needed before this proc)
6468                              <1>       ;
6469                              <1>       ; INPUT ->
6470                              <1>       ;     EAX = PHYSICAL ADDRESS (beginning address)
6471                              <1>       ;     EBX = VIRTUAL ADDRESS (beginning address)
6472                              <1>       ;     ECX = byte count (>=4096)
6473                              <1>       ;     [u.pgdir] = user's page directory
6474                              <1>       ;
6475                              <1>       ;     Note: All addresses are (must be) already adjusted
6476                              <1>       ;     to page      borders, otherwise, lower 12bits of addresses
6477                              <1>       ;     and byte count would be truncated.
```

```
6478                                 <1>       ;
6479                                 <1>       ; OUTPUT ->
6480                                 <1>       ;     none
6481                                 <1>       ;
6482                                 <1>       ;     CF = 1 -> insufficient memory error
6483                                 <1>       ;
6484                                 <1>       ; Note: All pages will be allocated in physical page order
6485                                 <1>       ;     from the beginning page address.
6486                                 <1>       ;     * A new page table will be added to the page dir
6487                                 <1>       ;       when the requested PDE is invalid.
6488                                 <1>       ;     * Those pages will not be added to swap queue
6489                                 <1>       ;       because main purpose of this allocation is to
6490                                 <1>       ;       set a direct memory access (DMA controller) buffer.
6491                                 <1>       ;      (Swapping out a page in a DMA buffer would be wrong!)
6492                                 <1>       ;     * Previous content of page tables (PTEs) would be
6493                                 <1>       ;       (should be) deallocated before entering this
6494                                 <1>       ;       procedure. So, new page table entries (PTEs)
6495                                 <1>       ;       directly will be written without checking
6496                                 <1>       ;       their previous content.
6497                                 <1>       ;     * Only solution to increase free memory by removing
6498                                 <1>       ;       that non-swappable memory block is to terminate
6499                                 <1>       ;       the process or to wait until the process will
6500                                 <1>       ;       deallocate that memory block as itself. ('sysdalloc')
6501                                 <1>       ;       (No problem, if the process does not grab all of
6502                                 <1>       ;       -very big amount of- free memory by using
6503                                 <1>       ;       'sysalloc' system call!?)
6504                                 <1>       ;       (Even if the process has grabbed all of free memory,
6505                                 <1>       ;       no problem if the process is not running in
6506                                 <1>       ;       multitasking mode. No problem in multitasking
6507                                 <1>       ;       mode if there is not another process which is running
6508                                 <1>       ;       or waiting or sleeping for an event as it's pages
6509                                 <1>       ;       are swapped-out. But a new process can not start to
6510                                 <1>       ;       run if all of free memory has beeen allocated
6511                                 <1>       ;       by running processes. Deallocation -'sysdalloc'-
6512                                 <1>       ;       or terminate a running process is needed
6513                                 <1>       ;       in order to run a new process.)
6514                                 <1>       ;
6515                                 <1>       ; Modified Registers -> EAX, EDX, ESI, EDI, EBX, ECX, EBP
6516                                 <1>       ;
6517                                 <1>
6518                                 <1>       ; 01/05/2017
6519 000063CA 662500F0              <1>       and   ax, ~PAGE_OFF
6520 000063CE 6681E300F0            <1>       and   bx, ~PAGE_OFF
6521                                 <1>       ; 02/05/2017
6522 000063D3 BD00F0FFFF            <1>       mov   ebp, 0FFFFF000h ; 4 Giga Bytes - 4096 Bytes (for Stack)
6523 000063D8 C1E90C                <1>       shr   ecx, PAGE_SHIFT ; page count
6524 000063DB 83F901                <1>       cmp   ecx, 1
6525 000063DE 7251                  <1>       jb    short a_u_im_retn
6526 000063E0 89C2                  <1>       mov   edx, eax
6527 000063E2 01CA                  <1>       add   edx, ecx
6528 000063E4 724B                  <1>       jc    short a_u_im_retn
6529 000063E6 39D5                  <1>       cmp   ebp, edx
6530 000063E8 7247                  <1>       jb    short a_u_im_retn
6531 000063EA 89DA                  <1>       mov   edx, ebx
6532 000063EC 81C200004000          <1>       add   edx, CORE
6533 000063F2 723D                  <1>       jc    short a_u_im_retn
6534 000063F4 01CA                  <1>       add   edx, ecx
6535 000063F6 7239                  <1>       jc    short a_u_im_retn
6536 000063F8 39D5                  <1>       cmp   ebp, edx
6537 000063FA 7235                  <1>       jb    short a_u_im_retn
6538                                 <1>       ;
6539 000063FC 89C5                  <1>       mov   ebp, eax ; physical address
6540 000063FE 89DE                  <1>       mov   esi, ebx
6541 00006400 81C600004000          <1>       add   esi, CORE ; start of user's memory (4M)
6542 00006406 C1EE0C                <1>       shr   esi, PAGE_SHIFT ; higher 20 bits of the linear address
6543                                 <1>       ;shr  ecx, PAGE_SHIFT ; page count
6544 00006409 8B1D[B8030300]        <1>       mov   ebx, [u.pgdir] ; physical addr of user's page dir
6545 0000640F 89F7                  <1>       mov   edi, esi
6546 00006411 81E7FF030000          <1>       and   edi, PTE_MASK ; PTE entry index in the page table
6547 00006417 57                    <1>       push  edi  ; * ; PTE index (in page directory)
6548 00006418 C1EE0A                <1>       shr   esi, PAGE_D_SHIFT - PAGE_SHIFT ; 22-12=10
6549 0000641B 89F2                  <1>       mov   edx, esi
6550                                 <1>       ; EDX = PDE index
6551 0000641D C1E602                <1>       shl   esi, 2 ; convert PDE index to dword offset
6552 00006420 01DE                  <1>       add   esi, ebx ; add page directory address
6553                                 <1> a_u_pd_0:
6554 00006422 AD                    <1>       lodsd
6555                                 <1>       ;
6556 00006423 89F3                  <1>       mov   ebx, esi ; next PDE address
6557                                 <1>       ;
6558 00006425 A801                  <1>       test  al, PDE_A_PRESENT ; bit 0, present flag (must be 1)
6559 00006427 7513                  <1>       jnz   short a_u_pd_2
6560                                 <1>       ;
6561                                 <1>       ; empty PDE (it does not point to valid page table address)
6562 00006429 E847F6FFFF            <1>       call  allocate_page  ; (allocate a new page table)
6563 0000642E 7302                  <1>       jnc   short a_u_pd_1 ; OK... now, we have a new page table.
6564                                 <1>       ; cf = 1
6565                                 <1>       ; There is not a free memory page to allocate a new page table !!!
6566 00006430 5E                    <1>       pop   esi ; *
6567                                 <1> a_u_im_retn:
6568 00006431 C3                    <1>       retn  ; return to 'sysalloc' with 'insufficient memory' error
6569                                 <1>       ;
6570                                 <1> a_u_pd_1: ; clear the new page table content
6571                                 <1>       ; EAX = Physical (base) address of the new page table
6572 00006432 E8AFF6FFFF            <1>       call  clear_page ; Clear page content
6573                                 <1>       ;
6574 00006437 0C07                  <1>       or    al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER
6575                                 <1>             ; set bit 0, bit 1 and bit 2 to 1
6576                                 <1>             ; (present, writable, user)
6577 00006439 8946FC                <1>       mov   [esi-4], eax
6578                                 <1> a_u_pd_2:
6579 0000643C 662500F0              <1>       and   ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
6580                                 <1>       ; EAX = PHYSICAL (flat) ADDRESS OF THE PAGE TABLE
6581 00006440 8B3C24                <1>       mov   edi, [esp] ; *
6582                                 <1>       ; EDI = PTE index (of page directory)
```

```
                        <1>       ;and   edi, PTE_MASK ; PTE entry index in the page table
                        <1>       ; EBX = next PDE address
00006443 89FE           <1>       mov    esi, edi ; PTE index in page table (0-1023)
00006445 C1E702         <1>       shl    edi, 2 ; convert PTE index to dword offset
00006448 01C7           <1>       add    edi, eax ; now, edi points to requested PTE
                        <1> a_u_pt_0:
                        <1>       ; 02/05/2017
0000644A 8B07           <1>       mov    eax, [edi]
                        <1>       ;
0000644C A801           <1>       test   al, PTE_A_PRESENT ; bit 0, present flag (must be 1)
0000644E 7445           <1>       jz     short a_u_pt_1
                        <1>       ;
00006450 A802           <1>       test   al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
                        <1>                                ; (must be 1)
00006452 7550           <1>       jnz    short a_u_pt_3
                        <1>       ; Read only -duplicated- page (belongs to a parent or a child)
00006454 66A90002       <1>       test  ax, PTE_DUPLICATED ; Was this page duplicated
                        <1>                                ; as child's page ?
00006458 7455           <1>       jz     short a_u_pt_4   ; Clear PTE but don't deallocate the page!
                        <1>       ;
                        <1>       ; check the parent's PTE value is read only & same page or not..
                        <1>       ; EDX = page directory entry index (0-1023)
0000645A 52             <1>       push   edx ; **
0000645B 53             <1>       push   ebx ; ***
                        <1>       ; ESI = page table entry index (0-1023)
                        <1>       ;push  esi ; **** ; 20/05/2017
0000645C 8B1D[BC030300] <1>       mov    ebx, [u.ppgdir] ; page directory of the parent process
00006462 66C1E202       <1>       shl    dx, 2 ; *4
00006466 01D3           <1>       add    ebx, edx ; PTE address,0 (for the parent)
00006468 8B13           <1>       mov    edx, [ebx] ; page table address
0000646A F6C201         <1>       test   dl, PDE_A_PRESENT ; present (valid) or not ?
0000646D 7433           <1>       jz     short a_u_pt_2      ; parent process does not use this page
0000646F 6681E200F0     <1>       and    dx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
00006474 66C1E602       <1>       shl    si, 2 ; *4
                        <1>       ; ESI = page table entry offset (0-4092)
00006478 01D6           <1>       add    esi, edx     ; PTE address (for the parent)
0000647A 8B1E           <1>       mov    ebx, [esi]
0000647C F6C301         <1>       test   bl, PTE_A_PRESENT ; present or not ?
0000647F 7421           <1>       jz     short a_u_pt_2      ; parent process does not use this page
00006481 662500F0       <1>       and    ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
00006485 6681E300F0     <1>       and    bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
0000648A 39D8           <1>       cmp    eax, ebx     ; parent's and child's pages are same ?
0000648C 7514           <1>       jne    short a_u_pt_2      ; not same page
                        <1>                                ; deallocate the child's page
0000648E 800E02         <1>       or    byte [esi], PTE_A_WRITE ; convert to writable page (parent)
                        <1>       ;pop   esi ; **** ; 20/05/2017
00006491 5B             <1>       pop    ebx ; ***
00006492 5A             <1>       pop    edx ; **
00006493 EB1A           <1>       jmp    short a_u_pt_4
                        <1> a_u_pt_1:
00006495 09C0           <1>       or     eax, eax     ; swapped page ?
00006497 7416           <1>       jz     short a_u_pt_4      ; no
                        <1>                                ; yes
00006499 D1E8           <1>       shr    eax, 1
0000649B E8C2F9FFFF     <1>       call   unlink_swap_block ; Deallocate swapped page block
                        <1>                                ; on the swap disk (or in file)
000064A0 EB0D           <1>       jmp    short a_u_pt_4
                        <1> a_u_pt_2:
                        <1>       ;pop   esi ; **** ; 20/05/2017
000064A2 5B             <1>       pop    ebx ; ***
000064A3 5A             <1>       pop    edx ; **
                        <1> a_u_pt_3:
000064A4 66A90004       <1>       test   ax, PTE_SHARED      ; shared or direct memory access indicator
000064A8 7505           <1>       jnz    short a_u_pt_4      ; AVL bit 1 = 1, do not deallocate this page!
                        <1>       ;
                        <1>       ;and   ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
000064AA E88EF7FFFF     <1>       call   deallocate_page ; set the mem allocation bit of this page
                        <1>       ;
                        <1> a_u_pt_4:
000064AF 89E8           <1>       mov    eax, ebp ; physical address
000064B1 0C07           <1>       or     al, PTE_A_PRESENT + PTE_A_WRITE + PTE_A_USER ; 04/03/2017
000064B3 AB             <1>       stosd
000064B4 5E             <1>       pop    esi ; * ; 20/05/2017
000064B5 49             <1>       dec    ecx ; remain page count
000064B6 7417           <1>       jz     short a_u_pd_5
000064B8 81C500100000   <1>       add    ebp, PAGE_SIZE
000064BE 46             <1>       inc    esi ; next PTE (index)
                        <1>       ; 20/05/2017
                        <1>       ;cmp   esi, PAGE_SIZE/4 ; 1024
                        <1>       ;jb    short a_u_pt_0
000064BF 6681E6FF03     <1>       and    si, PTE_MASK ; 3FFh (0 to 1023)
000064C4 56             <1>       push   esi ; *
000064C5 7583           <1>       jnz    short a_u_pt_0 ; > 0 (<1024)
                        <1> a_u_pd_3:
000064C7 42             <1>       inc    edx
                        <1> ;     cmp    edx, 1024
                        <1> ;     jnb    short a_u_pd_4 ; 02/05/2017 (error!, ecx > 0)
000064C8 89DE           <1>       mov    esi, ebx ; the next PDE address
000064CA E953FFFFFF     <1>       jmp    a_u_pd_0
                        <1> a_u_pd_4:
                        <1>       ; 02/05/2017
                        <1> ;     stc
                        <1> a_u_pd_5:
                        <1>       ; 20/05/2017
                        <1>       ;pop   edi ; *
000064CF C3             <1>       retn
                        <1>
                        <1> allocate_lfb_pages_for_kernel:
                        <1>       ; 15/12/2020
                        <1>       ; 14/12/2020 - TRDOS 386 v2.0.3
                        <1>       ; Set kernel page tables for linear frame buffer
                        <1>       ; (this procedure will be called by kernel only)
                        <1>       ;
                        <1>       ; Input:
                        <1>       ;      [LFB_ADDR] = linear frame buffer base address
```

```
6688                                    <1>     ;       [LFB_SIZE] = linear frame buffer size in bytes
6689                                    <1>     ; Output:
6690                                    <1>     ;       none
6691                                    <1>     ;       cf = 1 -> error
6692                                    <1>     ;
6693                                    <1>     ; Modified registers: eax, ecx, edx, edi
6694                                    <1>
6695 000064D0 8B3D[14120300]           <1>     mov     edi, [LFB_ADDR]
6696 000064D6 8B15[18120300]           <1>     mov     edx, [LFB_SIZE]
6697                                    <1>
6698 000064DC C1EF16                   <1>     shr     edi, 22       ; convert address to page number
6699                                    <1>                           ; and then convert it to PDE entry offset
6700                                    <1>                           ; (1 PDE is for 4MB, 22 bit shift)
6701                                    <1>
6702 000064DF 66C1E702                 <1>     shl     di, 2  ; * 4 for offset
6703                                    <1>
6704                                    <1>     ;add    edx, 4095
6705 000064E3 C1EA0C                   <1>     shr     edx, 12       ; convert LFB size to LFB page count
6706                                    <1>
6707 000064E6 89D1                     <1>     mov     ecx, edx ; * ; LFB page count
6708                                    <1>
6709 000064E8 81C1FF030000             <1>     add     ecx, 1023 ; page count + 1023
6710 000064EE C1E90A                   <1>     shr     ecx, 10 ; convert to page directory entry count
6711                                    <1>                           ; (page table count)
6712 000064F1 51                       <1>     push    ecx ; **
6713 000064F2 C1E10C                   <1>     shl     ecx, 12 ; convert to byte count
6714                                    <1>
6715 000064F5 31C0                     <1>     xor     eax, eax ; first available pages
6716                                    <1>
6717                                    <1>     ; allocate contiguous memory block for these kernel pages
6718                                    <1>
6719 000064F7 E89BFAFFFF               <1>     call    allocate_memory_block
6720                                    <1>     ; eax = start address of (contiguous) memory block
6721 000064FC 59                       <1>     pop     ecx ; ** ; PDE count
6722 000064FD 7301                     <1>     jnc     short a_lfb_k_1
6723                                    <1>     ; error (cf=1)
6724 000064FF C3                       <1>     retn
6725                                    <1> a_lfb_k_1:
6726                                    <1>     ; Allocate (new) page tables in kernel's page directory
6727 00006500 51                       <1>     push    ecx ; PDE (page table) count
6728 00006501 50                       <1>     push    eax ; start address of contiguous memory pages
6729                                    <1>                 ; (at page boundary)
6730                                    <1>     ; edi = 1st page directory entry offset
6731 00006502 033D[A8800100]           <1>     add     edi, [k_page_dir] ; Kernel's Page Dir Address
6732                                    <1> a_lfb_k_2:
6733 00006508 660D0304                 <1>     or      ax, PDE_A_PRESENT + PDE_A_WRITE + PDE_EXTERNAL
6734                                    <1>                         ; supervisor + read&write + present
6735                                    <1>                         ; + external memory block (LFB)
6736 0000650C AB                       <1>     stosd
6737 0000650D 0500100000               <1>     add     eax, 4096
6738 00006512 E2F4                     <1>     loop    a_lfb_k_2
6739                                    <1>
6740 00006514 5F                       <1>     pop     edi ; start addr of contiguous memory pages
6741 00006515 59                       <1>     pop     ecx ; page table (PDE) count
6742                                    <1>
6743                                    <1>     ; Allocate pages in (new) kernel page tables
6744                                    <1>
6745                                    <1>     ; (Note: page tables are contiguous in pyhsical memory)
6746 00006516 C1E10A                   <1>     shl     ecx, 10 ; * 1024, convert to (total) PTE count
6747                                    <1>
6748 00006519 A1[14120300]             <1>     mov     eax, [LFB_ADDR]
6749                                    <1>     ; edx = LFB page count
6750                                    <1>     ;and    ax, ~4095  ; lw of LFB address is 0
6751                                    <1> a_lfb_k_3:
6752 0000651E 660D0304                 <1>     or      ax, PTE_A_PRESENT + PTE_A_WRITE + PTE_EXTERNAL
6753                                    <1>                         ; supervisor + read&write + present
6754                                    <1>                         ; + external memory block (LFB)
6755 00006522 AB                       <1>     stosd
6756 00006523 4A                       <1>     dec     edx
6757 00006524 7408                     <1>     jz      short a_lfb_k_4 ; LFB size has been completed (!?)
6758 00006526 0500100000               <1>     add     eax, 4096
6759 0000652B E2F1                     <1>     loop    a_lfb_k_3
6760                                    <1>
6761 0000652D C3                       <1>     retn
6762                                    <1>
6763                                    <1> a_lfb_k_4:
6764                                    <1>     ; clear PTEs for empty/free pages
6765                                    <1>     ;       (if there are after LFB !?)
6766 0000652E 31C0                     <1>     xor     eax, eax ; clear page table entry (empty)
6767 00006530 F3AB                     <1>     rep     stosd
6768 00006532 C3                       <1>     retn
6769                                    <1>
6770                                    <1> ;deallocate_lfb_pages_for_kernel:
6771                                    <1>     ; 15/12/2020
6772                                    <1>     ; 14/12/2020 - TRDOS 386 v2.0.3
6773                                    <1>     ; Reset/Release kernel page tables
6774                                    <1>     ;       which are used for linear frame buffer
6775                                    <1>     ; (this procedure will be called by kernel only)
6776                                    <1>     ;
6777                                    <1>     ; Input:
6778                                    <1>     ;       [LFB_ADDR] = linear frame buffer base address
6779                                    <1>     ;       [FFB_SIZE] = linear frame buffer size in bytes
6780                                    <1>     ; Output:
6781                                    <1>     ;       none
6782                                    <1>     ;
6783                                    <1>     ; Modified registers: eax, ecx, edi
6784                                    <1>
6785                                    <1>     ;mov    edi, [LFB_ADDR]
6786                                    <1>     ;mov    ecx, [LFB_SIZE]
6787                                    <1>     ;
6788                                    <1>     ;shr    edi, 22       ; convert address to page number
6789                                    <1>     ;             ; and then convert it to PDE entry offset
6790                                    <1>     ;             ; (1 PDE is for 4MB, 22 bit shift)
6791                                    <1>     ;
6792                                    <1>     ;shl    di, 2  ; * 4 for offset
```

```
6793                                <1>       ;
6794                                <1>       ;;add  ecx, 4095
6795                                <1>       ;shr   ecx, 12      ; convert LFB size to page count
6796                                <1>       ;
6797                                <1>       ;add   ecx, 1023 ; page count + 1023
6798                                <1>       ;shr   ecx, 10 ; convert to page directory entry count
6799                                <1>       ;            ; (page table count)
6800                                <1>       ;push  ecx ; *
6801                                <1>       ;shl   ecx, 12 ; convert to byte count
6802                                <1>       ;
6803                                <1>       ;xor   eax, eax ; first available pages
6804                                <1>       ;
6805                                <1>       ;; deallocate contiguous memory block for kernel pages
6806                                <1>       ;
6807                                <1>       ;call  deallocate_memory_block
6808                                <1>       ;
6809                                <1>       ;pop   ecx ; * ; PDE count
6810                                <1>       ;
6811                                <1>       ;; Release/Free PDEs (page tables) in kernel's page dir
6812                                <1>       ;; edi = 1st page directory entry offset
6813                                <1>       ;add   edi, [k_page_dir] ; Kernel's Page Dir Address
6814                                <1>       ;sub   eax, eax ; clear (also invalidate)
6815                                <1>       ;rep   stosd
6816                                <1>       ;
6817                                <1>       ;retn
6818                                <1>
6819                                <1> ; /// End Of MEMORY MANAGEMENT FUNCTIONS ///
6820                                <1>
6821                                <1> ;; Data:
6822                                <1>
6823                                <1> ; 09/03/2015
6824                                <1> ;swpq_count: dw 0 ; count of pages on the swap que
6825                                <1> ;swp_drv:    dd 0 ; logical drive description table address of the swap drive/disk
6826                                <1> ;swpd_size:  dd 0 ; size of swap drive/disk (volume) in sectors (512 bytes).

6827                                <1> ;swpd_free:  dd 0 ; free page blocks (4096 bytes) on swap disk/drive (logical)
6828                                <1> ;swpd_next:  dd 0 ; next free page block
6829                                <1> ;swpd_last:  dd 0 ; last swap page block
2915                                    %include 'timer.s'   ; 17/01/2015
2916                                <1> ; ****************************************************************************
2917                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.4 - timer.s
2918                                <1> ; ----------------------------------------------------------------------------
2919                                <1> ; Last Update: 18/04/2021
2920                                <1> ; ----------------------------------------------------------------------------
2921                                <1> ; Beginning: 17/01/2016
2922                                <1> ; ----------------------------------------------------------------------------
2923                                <1> ; Assembler: NASM version 2.11 (trdos386.s)
2924                                <1> ; ----------------------------------------------------------------------------
2925                                <1> ; Turkish Rational DOS
2926                                <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
2927                                <1> ;
2928                                <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
2929                                <1> ;
2930                                <1> ; Derived from 'IBM PC-AT' BIOS source code (1985)
2931                                <1> ; ****************************************************************************
2932                                <1>
2933                                <1> ; TRDOS 386  (TRDOS v2.0) Kernel - TIMER & REAL TIME CLOCK (BIOS) FUNCTIONS
2934                                <1>
2935                                <1> ; IBM PC-AT BIOS Source Code ('BIOS2.ASM')
2936                                <1> ; TITLE BIOS2 ---- 06/10/85 BIOS INTERRUPT ROUTINES
2937                                <1>
2938                                <1> ;
2939                                <1> ; ///////// TIMER (& REAL TIME CLOCK) FUNCTIONS ///////////////
2940                                <1>
2941                                <1> int1Ah:
2942                                <1>       ; 29/01/2016
2943                                <1>       ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
2944 00006533 9C                  <1>       pushfd
2945 00006534 0E                  <1>       push   cs
2946 00006535 E801000000          <1>       call   TIME_OF_DAY_1
2947 0000653A C3                  <1>       retn
2948                                <1>
2949                                <1> ;--- INT  1A H -- (TIME OF DAY) ------------------------------------------------
2950                                <1> ;       THIS BIOS ROUTINE ALLOWS THE CLOCKS TO BE SET OR READ                :
2951                                <1> ;                                                                            :
2952                                <1> ; PARAMETERS:                                                                :
2953                                <1> ;     (AH) = 00H  READ THE CURRENT SETTING AND RETURN WITH,                  :
2954                                <1> ;                       (CX) = HIGH PORTION OF COUNT                         :
2955                                <1> ;                       (DX) = LOW PORTION OF COUNT                          :
2956                                <1> ;                       (AL) = 0 TIMER HAS NOT PASSED 24 HOURS SINCE LAST READ :
2957                                <1> ;                              1 IF ON ANOTHER DAY. (RESET TO ZERO AFTER READ) :
2958                                <1> ;                                                                            :
2959                                <1> ;     (AH) = 01H  SET THE CURRENT CLOCK USING,                               :
2960                                <1> ;                 (CX) = HIGH PORTION OF COUNT                               :
2961                                <1> ;                 (DX) = LOW PORTION OF COUNT.                               :
2962                                <1> ;                                                                            :
2963                                <1> ;                NOTE: COUNTS OCCUR AT THE RATE OF 1193180/65536 COUNTS/SECOND :
2964                                <1> ;                             (OR ABOUT 18.2 PER SECOND -- SEE EQUATES)      :
2965                                <1> ;                                                                            :
2966                                <1> ;     (AH) = 02H  READ THE REAL TIME CLOCK AND RETURN WITH,                  :
2967                                <1> ;                       (CH) = HOURS IN BCD (00-23)                          :
2968                                <1> ;                       (CL) = MINUTES IN BCD (00-59)                        :
2969                                <1> ;                       (DH) = SECONDS IN BCD (00-59)                        :
2970                                <1> ;                       (DL) = DAYLIGHT SAVINGS ENABLE (00-01)               :
2971                                <1> ;                                                                            :
2972                                <1> ;     (AH) = 03H  SET THE REAL TIME CLOCK USING,                             :
2973                                <1> ;                       (CH) = HOURS IN BCD (00-23)                          :
2974                                <1> ;                       (CL) = MINUTES IN BCD (00-59)                        :
2975                                <1> ;                       (DH) = SECONDS IN BCD (00-59)                        :
2976                                <1> ;                       (DL) = 01 IF DAYLIGHT SAVINGS ENABLE OPTION, ELSE 00.  :
2977                                <1> ;                                                                            :
2978                                <1> ;              NOTE: (DL) = 00 IF DAYLIGHT SAVINGS TIME ENABLE IS NOT ENABLED. :
2979                                <1> ;                    (DL) = 01 ENABLES TWO SPECIAL UPDATES THE LAST SUNDAY IN  :
2980                                <1> ;                  APRIL   (1:59:59 --> 3:00:00 AM) AND THE LAST SUNDAY IN     :
2981                                <1> ;                    OCTOBER (1:59:59 --> 1:00:00 AM) THE FIRST TIME.          :
```

```
                                <1> ;                                                                              :
                                <1> ;     (AH) = 04H  READ THE DATE FROM THE REAL TIME CLOCK AND RETURN WITH,       :
                                <1> ;                      (CH) = CENTURY IN BCD (19 OR 20)                        :
                                <1> ;                      (CL) = YEAR IN BCD (00-99)                              :
                                <1> ;                      (DH) = MONTH IN BCD (01-12)                             :
                                <1> ;                      (DL) = DAY IN BCD (01-31).                              :
                                <1> ;                                                                              :
                                <1> ;     (AH) = 05H  SET THE DATE INTO THE REAL TIME CLOCK USING,                 :
                                <1> ;                    (CH) = CENTURY IN BCD (19 OR 20)                          :
                                <1> ;                    (CL) = YEAR IN BCD (00-99)                                :
                                <1> ;                    (DH) = MONTH IN BCD (01-12)                               :
                                <1> ;                    (DL) = DAY IN BCD (01-31).                                :
                                <1> ;                                                                              :
                                <1> ;     (AH) = 06H  SET THE ALARM TO INTERRUPT AT SPECIFIED TIME,                :
                                <1> ;                    (CH) = HOURS IN BCD (00-23 (OR FFH))                      :
                                <1> ;                    (CL) = MINUTES IN BCD (00-59 (OR FFH))                    :
                                <1> ;                    (DH) = SECONDS IN BCD (00-59 (OR FFH))                    :
                                <1> ;                                                                              :
                                <1> ;     (AH) = 07H  RESET THE ALARM INTERRUPT FUNCTION.                          :
                                <1> ;                                                                              :
                                <1> ; NOTES: FOR ALL RETURNS CY= 0 FOR SUCCESSFUL OPERATION.                       :
                                <1> ;        FOR (AH)= 2, 4, 6 - CARRY FLAG SET IF REAL TIME CLOCK NOT OPERATING.  :
                                <1> ;        FOR (AH)= 6 - CARRY FLAG SET IF ALARM ALREADY ENABLED.                :
                                <1> ;        FOR THE ALARM FUNCTION (AH = 6) THE USER MUST SUPPLY A ROUTINE AND    :
                                <1> ;         INTERCEPT THE CORRECT ADDRESS IN THE VECTOR TABLE FOR INTERRUPT 4AH.:
                                <1> ;         USE 0FFH FOR ANY "DO NOT CARE" POSITION FOR INTERVAL INTERRUPTS.     :
                                <1> ;        INTERRUPTS ARE DISABLED DURING DATA MODIFICATION.                     :
                                <1> ;        AH & AL ARE RETURNED MODIFIED AND NOT DEFINED EXCEPT WHERE INDICATED.:
                                <1> ;--------------------------------------------------------------------------------
                                <1>
                                <1> ; 15/01/2017
                                <1> ; 14/01/2017
                                <1> ; 07/01/2017
                                <1> ; 02/01/2017
                                <1> ; 29/05/2016
                                <1> ; 29/01/2016
                                <1> ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>
                                <1> ; 29/05/2016
                                <1> ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                                <1> int35h:  ; Date/Time functions
                                <1>
                                <1> TIME_OF_DAY_1:
                                <1>       ;sti                      ; INTERRUPTS BACK ON
                                <1>       ; 29/05/2016
0000653B 80642408FE             <1>       and   byte [esp+8], 11111110b   ; clear carry bit of eflags register
                                <1>       ;
00006540 80FC08                 <1>       cmp   ah, (RTC_TBE-RTC_TB)/4    ; CHECK IF COMMAND IN VALID RANGE (0-7)
00006543 F5                     <1>       cmc                       ; COMPLEMENT CARRY FOR ERROR EXIT
                                <1>       ; (*) jc short TIME_9           ; EXIT WITH CARRY = 1 IF NOT VALID
00006544 721A                   <1>       jc    short _TIME_9 ; 29/05/2016
                                <1>
00006546 1E                     <1>       push  ds
00006547 56                     <1>       push  esi
00006548 66BE1000               <1>       mov   si, KDATA           ; kernel data segment
0000654C 8EDE                   <1>       mov   ds, si
                                <1>
                                <1>       ;;15/01/2017
                                <1>       ; 14/01/2017
                                <1>       ; 02/01/2017
                                <1>       ;;mov  byte [intflg], 35h  ; date & time interrupt
                                <1>       ;sti
                                <1>       ;
0000654E C0E402                 <1>       shl   ah, 2               ; convert function to dword offset
00006551 0FB6F4                 <1>       movzx esi, ah                   ; PLACE INTO ADDRESSING REGISTER
                                <1>       ;cli                      ; NO INTERRUPTS DURING TIME FUNCTIONS
00006554 FF96[66650000]         <1>       call  [esi+RTC_TB]        ; VECTOR TO FUNCTION REQUESTED WITH CY=0
                                <1>                                 ; RETURN WITH CARRY FLAG SET FOR RESULT
                                <1>       ;sti                      ; INTERRUPTS BACK ON
0000655A B400                   <1>       mov   ah, 0               ; CLEAR (AH) TO ZERO
0000655C 5E                     <1>       pop   esi                 ; RECOVER USERS REGISTER
0000655D 1F                     <1>       pop   ds                  ; RECOVER USERS SEGMENT SELECTOR
                                <1>
                                <1>       ;;15/01/2017
                                <1>       ; 02/01/2017
                                <1>       ;;mov  byte [ss:intflg], 0 ; 07/01/2017
                                <1>
                                <1> ;TIME_9:
                                <1>                                 ; RETURN WITH CY= 0 IF NO ERROR
                                <1>       ; (*) 29/05/2016
                                <1>       ; (*) retf 4 ; skip eflags on stack
0000655E 7305                   <1>       jnc   short _TIME_10
                                <1> _TIME_9:
                                <1>       ; 29/05/2016 -set carry flag on stack-
                                <1>       ; [esp] = EIP
                                <1>       ; [esp+4] = CS
                                <1>       ; [esp+8] = E-FLAGS
00006560 804C240801             <1>       or    byte [esp+8], 1     ; set carry bit of eflags register
                                <1>       ; [esp+12] = ESP (user)
                                <1>       ; [esp+16] = SS (User)
                                <1> _TIME_10:
00006565 CF                     <1>       iretd
                                <1>
                                <1>       ; (*) 29/05/2016 - 'ref 4' intruction causes to stack fault
                                <1>       ; (OUTER-PRIVILEGE-LEVEL)
                                <1>       ; INTEL 80386 PROGRAMMER'S REFERENCE MANUAL 1986
                                <1>       ; // RETF instruction:
                                <1>       ;
                                <1>       ; IF OperandMode=32 THEN
                                <1>       ;    Load CS:EIP from stack;
                                <1>       ;    Set CS RPL to CPL;
                                <1>       ;    Increment eSP by 8 plus the immediate offset if it exists;
                                <1>       ;    Load SS:eSP from stack;
                                <1>       ; ELSE (* OperandMode=16 *)
                                <1>       ;    Load CS:IP from stack;
```

```
                        <1>       ;    Set CS RPL to CPL;
                        <1>       ;    Increment eSP by 4 plus the immediate offset if it exists;
                        <1>       ;    Load SS:eSP from stack;
                        <1>       ; FI;
                        <1>       ;
                        <1>       ; //
                        <1>                                 ; ROUTINE VECTOR TABLE (AH)=
                        <1> RTC_TB:
00006566 [86650000]     <1>       dd     RTC_00             ; 0 = READ CURRENT CLOCK COUNT
0000656A [99650000]     <1>       dd     RTC_10             ; 1 = SET CLOCK COUNT
0000656E [A7650000]     <1>       dd     RTC_20             ; 2 = READ THE REAL TIME CLOCK TIME
00006572 [D6650000]     <1>       dd     RTC_30             ; 3 = SET REAL TIME CLOCK TIME
00006576 [18660000]     <1>       dd     RTC_40             ; 4 = READ THE REAL TIME CLOCK DATE
0000657A [45660000]     <1>       dd     RTC_50             ; 5 = SET REAL TIME CLOCK DATE
0000657E [92660000]     <1>       dd     RTC_60             ; 6 = SET THE REAL TIME CLOCK ALARM
00006582 [E5660000]     <1>       dd     RTC_70             ; 7 = RESET ALARM
                        <1>
                        <1> RTC_TBE      equ    $
                        <1>
                        <1> RTC_00:                         ; READ TIME COUNT
00006586 A0[2C810100]   <1>       mov    al, [TIMER_OFL]     ; GET THE OVERFLOW FLAG
0000658B C605[2C810100]00 <1>     mov    byte [TIMER_OFL], 0 ; AND THEN RESET THE OVERFLOW FLAG
00006592 8B0D[28810100] <1>       mov   ecx, [TIMER_LH]      ; GET COUNT OF TIME
00006598 C3             <1>       retn
                        <1>
                        <1> RTC_10:                         ; SET TIME COUNT
00006599 890D[28810100] <1>       mov   [TIMER_LH], ecx     ; SET TIME COUNT
0000659F C605[2C810100]00 <1>     mov    byte [TIMER_OFL], 0 ; RESET OVERFLOW FLAG
000065A6 C3             <1>       retn                      ; RETURN WITH NO CARRY
                        <1>
                        <1> RTC_20:                         ; GET RTC TIME
000065A7 E8C8010000     <1>       call   UPD_IPR            ; CHECK FOR UPDATE IN PROCESS
000065AC 7227           <1>       jc     short RTC_29       ; EXIT IF ERROR (CY= 1)
                        <1>
000065AE B000           <1>       mov    al, CMOS_SECONDS    ; SET ADDRESS OF SECONDS
000065B0 E8F5010000     <1>       call   CMOS_READ          ; GET SECONDS
000065B5 88C6           <1>       mov    dh, al             ; SAVE
000065B7 B00B           <1>       mov    al, CMOS_REG_B     ; ADDRESS ALARM REGISTER
000065B9 E8EC010000     <1>       call   CMOS_READ          ; READ CURRENT VALUE OF DSE BIT
000065BE 2401           <1>       and    al, 00000001b      ; MASK FOR VALID DSE BIT
000065C0 88C2           <1>       mov    dl, al             ; SET [DL] TO ZERO FOR NO DSE BIT
000065C2 B002           <1>       mov    al, CMOS_MINUTES   ; SET ADDRESS OF MINUTES
000065C4 E8E1010000     <1>       call   CMOS_READ          ; GET MINUTES
000065C9 88C1           <1>       mov    cl, al             ; SAVE
000065CB B004           <1>       mov   al, CMOS_HOURS      ; SET ADDRESS OF HOURS
000065CD E8D8010000     <1>       call   CMOS_READ          ; GET HOURS
000065D2 88C5           <1>       mov    ch, al             ; SAVE
000065D4 F8             <1>       clc                       ; SET CY= 0
                        <1> RTC_29:
000065D5 C3             <1>       retn                      ; RETURN WITH RESULT IN CARRY FLAG
                        <1>
                        <1> RTC_30:                         ; SET RTC TIME
000065D6 E899010000     <1>       call   UPD_IPR            ; CHECK FOR UPDATE IN PROCESS
000065DB 7305           <1>       jnc    short RTC_35       ; GO AROUND IF CLOCK OPERATING
000065DD E8AD010000     <1>       call   RTC_STA            ; ELSE TRY INITIALIZING CLOCK
                        <1> RTC_35:
000065E2 88F4           <1>       mov    ah, dh             ; GET TIME BYTE - SECONDS
000065E4 B000           <1>       mov    al, CMOS_SECONDS   ; ADDRESS SECONDS
000065E6 E8D6010000     <1>       call   CMOS_WRITE         ; UPDATE SECONDS
000065EB 88CC           <1>       mov    ah, cl             ; GET TIME BYTE - MINUTES
000065ED B002           <1>       mov    al, CMOS_MINUTES   ; ADDRESS MINUTES
000065EF E8CD010000     <1>       call   CMOS_WRITE         ; UPDATE MINUTES
000065F4 88EC           <1>       mov    ah, ch             ; GET TIME BYTE - HOURS
000065F6 B004           <1>       mov    al, CMOS_HOURS     ; ADDRESS HOURS
000065F8 E8C4010000     <1>       call   CMOS_WRITE         ; UPDATE ADDRESS
                        <1>       ;mov   al, CMOS_REG_B     ; ADDRESS ALARM REGISTER
                        <1>       ;mov   ah, al
000065FD 66B80B0B       <1>       mov    ax, CMOS_REG_B * 257
00006601 E8A4010000     <1>       call   CMOS_READ          ; READ CURRENT TIME
00006606 2462           <1>       and    al, 01100010b      ; MASK FOR VALID BIT POSITIONS
00006608 0C02           <1>       or     al, 00000010b      ; TURN ON 24 HOUR MODE
0000660A 80E201         <1>       and    dl, 00000001b      ; USE ONLY THE DSE BIT
0000660D 08D0           <1>       or     al, dl             ; GET DAY LIGHT SAVINGS TIME BIT (OSE)
0000660F 86E0           <1>       xchg   ah, al             ; PLACE IN WORK REGISTER AND GET ADDRESS
00006611 E8AB010000     <1>       call   CMOS_WRITE         ; SET NEW ALARM SITS
00006616 F8             <1>       clc                       ; SET CY= 0
00006617 C3             <1>       retn                      ; RETURN WITH CY= 0
                        <1>
                        <1> RTC_40:                         ; GET RTC DATE
00006618 E857010000     <1>       call   UPD_IPR            ; CHECK FOR UPDATE IN PROCESS
0000661D 7225           <1>       jc     short RTC_49       ; EXIT IF ERROR (CY= 1)
                        <1>
0000661F B007           <1>       mov    al, CMOS_DAY_MONTH ; ADDRESS DAY OF MONTH
00006621 E884010000     <1>       call   CMOS_READ          ; READ DAY OF MONTH
00006626 88C2           <1>       mov    dl, al             ; SAVE
00006628 B008           <1>       mov    al, CMOS_MONTH     ; ADDRESS MONTH
0000662A E87B010000     <1>       call   CMOS_READ          ; READ MONTH
0000662F 88C6           <1>       mov    dh, al             ; SAVE
00006631 B009           <1>       mov    al, CMOS_YEAR      ; ADDRESS YEAR
00006633 E872010000     <1>       call   CMOS_READ          ; READ YEAR
00006638 88C1           <1>       mov    cl, al             ; SAVE
0000663A B032           <1>       mov    al, CMOS_CENTURY   ; ADDRESS CENTURY LOCATION
0000663C E869010000     <1>       call   CMOS_READ          ; GET CENTURY BYTE
00006641 88C5           <1>       mov    ch, al             ; SAVE
00006643 F8             <1>       clc                       ; SET CY=0
                        <1> RTC_49:
00006644 C3             <1>       retn                      ; RETURN WITH RESULTS IN CARRY FLAG
                        <1>
                        <1> RTC_50:                         ; SET RTC DATE
00006645 E82A010000     <1>       call   UPD_IPR            ; CHECK FOR UPDATE IN PROCESS
0000664A 7305           <1>       jnc    short RTC_55       ; GO AROUND IF NO ERROR
0000664C E83E010000     <1>       call   RTC_STA            ; ELSE INITIALIZE CLOCK
                        <1> RTC_55:
00006651 66B80600       <1>       mov    ax, CMOS_DAY_WEEK  ; ADDRESS OF DAY OF WEEK BYTE
00006655 E867010000     <1>       call   CMOS_WRITE         ; LOAD ZEROS TO DAY OF WEEK
```

```
3192 0000665A 88D4              <1>       mov   ah, dl              ; GET DAY OF MONTH BYTE
3193 0000665C B007              <1>       mov   al, CMOS_DAY_MONTH  ; ADDRESS DAY OF MONTH BYTE
3194 0000665E E85E010000        <1>       call  CMOS_WRITE          ; WRITE OF DAY OF MONTH REGISTER
3195 00006663 88F4              <1>       mov   ah, dh              ; GET MONTH
3196 00006665 B008              <1>       mov   al, CMOS_MONTH      ; ADDRESS MONTH BYTE
3197 00006667 E855010000        <1>       call  CMOS_WRITE          ; WRITE MONTH REGISTER
3198 0000666C 88CC              <1>       mov   ah, cl              ; GET YEAR BYTE
3199 0000666E B009              <1>       mov   al, CMOS_YEAR       ; ADDRESS YEAR REGISTER
3200 00006670 E84C010000        <1>       call  CMOS_WRITE          ; WRITE YEAR REGISTER
3201 00006675 88EC              <1>       mov   ah, ch              ; GET CENTURY BYTE
3202 00006677 B032              <1>       mov   al, CMOS_CENTURY    ; ADDRESS CENTURY BYTE
3203 00006679 E843010000        <1>       call  CMOS_WRITE          ; WRITE CENTURY LOCATION
3204                            <1>       ;mov  al, CMOS_REG_B      ; ADDRESS ALARM REGISTER
3205                            <1>       ;mov  ah, al
3206 0000667E 66B80B0B          <1>       mov   ax, CMOS_REG_B * 257
3207 00006682 E823010000        <1>       call  CMOS_READ           ; READ WIRRENT SETTINGS
3208 00006687 247F              <1>       and   al, 07Fh            ; CLEAR 'SET BIT'
3209 00006689 86E0              <1>       xchg  ah, al              ; MOVE TO WORK REGISTER
3210 0000668B E831010000        <1>       call  CMOS_WRITE          ; AND START CLOCK UPDATING
3211 00006690 F8                <1>       clc                       ; SET CY= 0
3212 00006691 C3                <1>       retn                      ; RETURN CY=0
3213                            <1>
3214                            <1> RTC_60:                         ; SET RTC ALARM
3215 00006692 B00B              <1>       mov   al, CMOS_REG_B      ; ADDRESS ALARM
3216 00006694 E811010000        <1>       call  CMOS_READ           ; READ ALARM REGISTER
3217 00006699 A820              <1>       test  al, 20h             ; CHECK FOR ALARM ALREADY ENABLED
3218 0000669B F9                <1>       stc                       ; SET CARRY IN CASE OF ERROR
3219 0000669C 7542              <1>       jnz   short RTC_69        ; ERROR EXIT IF ALARM SET
3220 0000669E E8D1000000        <1>       call  UPD_IPR             ; CHECK FOR UPDATE IN PROCESS
3221 000066A3 7305              <1>       jnc   short RTC_65        ; SKIP INITIALIZATION IF NO ERROR
3222 000066A5 E8E5000000        <1>       call  RTC_STA             ; ELSE INITIALIZE CLOCK
3223                            <1> RTC_65:
3224 000066AA 88F4              <1>       mov   ah, dh              ; GET SECONDS BYTE
3225 000066AC B001              <1>       mov   al, CMOS_SEC_ALARM  ; ADDRESS THE SECONDS ALARM REGISTER
3226 000066AE E80E010000        <1>       call  CMOS_WRITE          ; INSERT SECONDS
3227 000066B3 88CC              <1>       mov   ah, cl              ; GET MINUTES PARAMETER
3228 000066B5 B003              <1>       mov   al, CMOS_MIN_ALARM  ; ADDRESS MINUTES ALARM REGISTER
3229 000066B7 E805010000        <1>       call  CMOS_WRITE          ; INSERT MINUTES
3230 000066BC 88EC              <1>       mov   ah, ch              ; GET HOURS PARAMETER
3231 000066BE B005              <1>       mov   al, CMOS_HR_ALARM   ; ADDRESS HOUR ALARM REGISTER
3232 000066C0 E8FC000000        <1>       call  CMOS_WRITE          ; INSERT HOURS
3233 000066C5 E4A1              <1>       in    al, INTB01          ; READ SECOND INTERRUPT MASK REGISTER
3234 000066C7 24FE              <1>       and   al, 0FEh            ; ENABLE ALARM TIMER BIT (CY= 0)
3235 000066C9 E6A1              <1>       out   INTB01, al          ; WRITE UPDATED MASK
3236                            <1>       ;mov  al, CMOS_REG_B      ; ADDRESS ALARM REGISTER
3237                            <1>       ;mov  ah, al
3238 000066CB 66B80B0B          <1>       mov   ax, CMOS_REG_B * 257
3239 000066CF E8D6000000        <1>       call  CMOS_READ           ; READ CURRENT ALARM REGISTER
3240 000066D4 247F              <1>       and   al, 07Fh            ; ENSURE SET BIT TURNED OFF
3241 000066D6 0C20              <1>       or    al, 20h                   ; TURN ON ALARM ENABLE
3242 000066D8 86E0              <1>       xchg  ah, al              ; MOVE MASK TO OUTPUT REGISTER
3243 000066DA E8E2000000        <1>       call  CMOS_WRITE          ; WRITE NEW ALARM MASK
3244 000066DF F8                <1>       clc                       ; SET CY= 0
3245                            <1> RTC_69:
3246 000066E0 66B80000          <1>       mov   ax, 0               ; CLEAR AX REGISTER
3247 000066E4 C3                <1>       retn                      ; RETURN WITH RESULTS IN CARRY FLAC
3248                            <1>
3249                            <1> RTC_70:                         ; RESET ALARM
3250                            <1>       ;mov  al, CMOS_REG_B      ; ADDRESS ALARM REGISTER
3251                            <1>       ;mov  ah, al
3252 000066E5 66B80B0B          <1>       mov   ax, CMOS_REG_B * 257 ; ADDRESS ALARM REGISTER (TO BOTH AH,AL)
3253 000066E9 E8BC000000        <1>       call  CMOS_READ           ; READ ALARM REGISTER
3254 000066EE 2457              <1>       and   al, 57h             ; TURN OFF ALARM ENABLE
3255 000066F0 86E0              <1>       xchg  ah, al              ; SAVE DATA AND RECOVER ADDRESS
3256 000066F2 E8CA000000        <1>       call  CMOS_WRITE          ; RESTORE NEW VALUE
3257 000066F7 F8                <1>       clc                       ; SET CY= 0
3258 000066F8 C3                <1>       retn                      ; RETURN WITH NO CARRY
3259                            <1>
3260                            <1> ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
3261                            <1>
3262                            <1> ;--- HARDWARE INT  70 H -- ( IRQ LEVEL  8) -------------------------------------
3263                            <1> ; ALARM INTERRUPT HANDLER (RTC)                                              :
3264                            <1> ;       THIS ROUTINE HANDLES THE PERIODIC AND ALARM INTERRUPTS FROM THE CMOS  :
3265                            <1> ;       TIMER. INPUT FREQUENCY IS 1.024 KHZ OR APPROXIMATELY 1024 INTERRUPTS  :
3266                            <1> ;       EVERY SECOND FOR THE PERIODIC INTERRUPT. FOR THE ALARM FUNCTION,      :
3267                            <1> ;       THE INTERRUPT WILL OCCUR AT THE DESIGNATED TIME.                      :
3268                            <1> ;                                                                             :
3269                            <1> ;       INTERRUPTS ARE ENABLED WHEN THE EVENT OR ALARM FUNCTION IS ACTIVATED. :
3270                            <1> ;       FOR THE EVENT INTERRUPT, THE HANDLER WILL DECREMENT THE WAIT COUNTER  :
3271                            <1> ;       AND WHEN IT EXPIRES WILL SET THE DESIGNATED LOCATION TO 80H. FOR      :
3272                            <1> ;       THE ALARM INTERRUPT. THE USER MUST PROVIDE A ROUTINE TO INTERCEPT     :
3273                            <1> ;       THE CORRECT ADDRESS FROM THE VECTOR TABLE INVOKED BY INTERRUPT 4AH     :
3274                            <1> ;       PRIOR TO SETTING THE REAL TIME CLOCK ALARM (INT 1AH, AH= 06H).        :
3275                            <1> ;-------------------------------------------------------------------------------
3276                            <1>
3277                            <1> RTC_A_INT: ; 07/01/2017
3278                            <1> ;RTC_INT:                       ; ALARM INTERRUPT
3279 000066F9 1E                <1>       push  ds                  ; LEAVE INTERRUPTS DISABLED
3280 000066FA 50                <1>       push  eax                 ; SAVE REGISTERS
3281 000066FB 57                <1>       push  edi
3282                            <1> RTC_I_1:                        ; CHECK FOR SECOND INTERRUPT
3283 000066FC 66B88C8B          <1>       mov   ax, 256*(CMOS_REG_B+NMI)+CMOS_REG_C+NMI ; ALARM AND STATUS
3284 00006700 E670              <1>       out   CMOS_PORT, al       ; WRITE ALARM FLAG MASK ADDRESS
3285 00006702 90                <1>       nop                       ; I/O DELAY
3286 00006703 EB00              <1>       jmp   short $+2
3287 00006705 E471              <1>       in    al, CMOS_DATA       ; READ AND RESET INTERRUPT REQUEST FLAGS
3288 00006707 A860              <1>       test  al, 01100000b       ; CHECK FOR EITHER INTERRUPT PENDING
3289 00006709 745B              <1>       jz    short  RTC_I_9      ; EXIT IF NOT A VALID RTC INTERRUPT
3290                            <1>
3291 0000670B 86E0              <1>       xchg  ah, al              ; SAVE FLAGS AND GET ENABLE ADDRESS
3292 0000670D E670              <1>       out   CMOS_PORT, al       ; WRITE ALARM ENABLE MASK ADDRESS
3293 0000670F 90                <1>       nop                       ; I/O DELAY
3294 00006710 EB00              <1>       jmp   short $+2
3295 00006712 E471              <1>       in    al, CMOS_DATA       ; READ CURRENT ALARM ENABLE MASK
3296 00006714 20E0              <1>       and   al, ah              ; ALLOW ONLY SOURCES THAT ARE ENABLED
```

```
3297 00006716 A840                <1>       test   al, 01000000b       ; CHECK FOR PERIODIC INTERRUPT
3298 00006718 7439                <1>       jz     short RTC_I_5       ; SKIP IF NOT A PERIODIC INTERRUPT
3299                              <1>
3300                              <1> ;-----      DECREMENT WAIT COUNT BY INTERRUPT INTERVAL
3301                              <1>
3302 0000671A 66BF1000            <1>       mov    di, KDATA           ; kernel data segment
3303 0000671E 8EDF                <1>       mov    ds, di
3304                              <1>
3305 00006720 812D[20810100]D003- <1>       sub    dword [RTC_LH], 976 ; DECREMENT COUNT BY 1/1024
3305 00006728 0000                <1>
3306 0000672A 7327                <1>       jnc    short RTC_I_5       ; SKIP TILL 32 BIT WORD LESS THAN ZERO
3307                              <1>
3308                              <1> ;-----      TURN OFF PERIODIC INTERRUPT ENABLE
3309                              <1>
3310                              <1>       ;push  ax                  ; SAVE INTERRUPT FLAG MASK
3311                              <1>       ; 18/04/2021
3312 0000672C 50                  <1>       push   eax
3313 0000672D 66B88B8B            <1>       mov    ax, 257*(CMOS_REG_B+NMI) ; INTERRUPT ENABLE REGISTER
3314 00006731 E670                <1>       out    CMOS_PORT, al       ; WRITE ADDRESS TO CMOS CLOCK
3315 00006733 90                  <1>       nop                        ; I/O DELAY
3316 00006734 EB00                <1>       jmp    short $+2
3317 00006736 E471                <1>       in     al, CMOS_DATA       ; READ CURRENT ENABLES
3318 00006738 24BF                <1>       and    al, 0BFh            ; TURN OFF PIE
3319 0000673A 86C4                <1>       xchg   al, ah              ; GET CMOS ADDRESS AND SAVE VALUE
3320 0000673C E670                <1>       out    CMOS_PORT, al       ; ADDRESS REGISTER B
3321 0000673E 86C4                <1>       xchg   al, ah              ; GET NEW INTERRUPT ENABLE MASK
3322 00006740 E671                <1>       out    CMOS_DATA, al       ; SET MASK IN INTERRUPT ENABLE REGISTER
3323 00006742 C605[24810100]00    <1>       mov    byte [RTC_WAIT_FLAG], 0 ; SET FUNCTION ACTIVE FLAG OFF
3324 00006749 8B3D[25810100]      <1>       mov    edi, [USER_FLAG]    ; SET UP (DS:DI) TO POINT TO USER FLAG
3325 0000674F C60780              <1>       mov    byte [edi], 80h     ; TURN ON USERS FLAG
3326                              <1>       ;pop   ax                  ; GET INTERRUPT SOURCE BACK
3327                              <1>       ; 18/04/2021
3328 00006752 58                  <1>       pop    eax
3329                              <1> RTC_I_5:
3330 00006753 A820                <1>       test   al, 00100000b       ; TEST FOR ALARM INTERRUPT
3331 00006755 740D                <1>       jz     short RTC_I_7       ; SKIP USER INTERRUPT CALL IF NOT ALARM
3332                              <1>
3333 00006757 B00D                <1>       mov    al, CMOS_REG_D      ; POINT TO DEFAULT READ ONLY REGISTER
3334 00006759 E670                <1>       out    CMOS_PORT, al       ; ENABLE NMI AND CMOS ADDRESS TO DEFAULT
3335 0000675B FB                  <1>       sti                        ; INTERRUPTS BACK ON NOW
3336 0000675C 52                  <1>       push   edx
3337 0000675D E836C00000          <1>       call   INT4Ah              ; TRANSFER TO USER ROUTINE
3338 00006762 5A                  <1>       pop    edx
3339 00006763 FA                  <1>       cli                        ; BLOCK INTERRUPT FOR RETRY
3340                              <1> RTC_I_7:                         ; RESTART ROUTINE TO HANDLE DELAYED
3341 00006764 EB96                <1>       jmp    short RTC_I_1       ;  ENTRY AND SECOND EVENT BEFORE DONE
3342                              <1>
3343                              <1> RTC_I_9:                         ; EXIT - NO PENDING INTERRUPTS
3344 00006766 B00D                <1>       mov    al, CMOS_REG_D      ; POINT TO DEFAULT READ ONLY REGISTER
3345 00006768 E670                <1>       out    CMOS_PORT, al       ; ENABLE NMI AND CMOS ADDRESS TO DEFAULT
3346 0000676A B020                <1>       mov    al, EOI             ; END OF INTERRUPT MASK TO 8259 - 2
3347 0000676C E6A0                <1>       out    INTB00, al          ; TO 8259 - 2
3348 0000676E E620                <1>       out    INTA00, al          ; TO 8259 - 1
3349 00006770 5F                  <1>       pop    edi                 ; RESTORE REGISTERS
3350 00006771 58                  <1>       pop    eax
3351 00006772 1F                  <1>       pop    ds
3352 00006773 CF                  <1>       iretd                      ; END OF INTERRUPT
3353                              <1>
3354                              <1>       ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
3355                              <1>       ; 22/08/2014 (Retro UNIX 386 v1)
3356                              <1>       ; IBM PC/AT BIOS source code ----- 10/06/85 (bios2.asm)
3357                              <1> UPD_IPR:                         ; WAIT TILL UPDATE NOT IN PROGRESS
3358 00006774 51                  <1>       push   ecx
3359                              <1>
3360                              <1>       ; 29/05/2016
3361 00006775 B968110000          <1>       mov    ecx, ((1984+244)*4)/2 ; AWARD BIOS 1999, ATIME.ASM
3362                              <1>                                   ; 'WAITCPU_CK_UD_STAT'
3363                              <1>                                   ; (244Us + 1984Us)
3364                              <1>                                   ; (assume each read takes
3365                              <1>                                   ;  2 microseconds).
3366                              <1>       ;mov   ecx, 65535
3367                              <1>             ;mov cx, 800         ; SET TIMEOUT LOOP COUNT (= 800)
3368                              <1> UPD_10:
3369 0000677A B00A                <1>       mov    al, CMOS_REG_A      ; ADDRESS STATUS REGISTER A
3370 0000677C FA                  <1>       cli                        ; NO TIMER INTERRUPTS DURING UPDATES
3371 0000677D E828000000          <1>       call   CMOS_READ           ; READ UPDATE IN PROCESS FLAG
3372 00006782 A880                <1>       test   al, 80h             ; IF UIP BIT IS ON ( CANNOT READ TIME )
3373 00006784 7406                <1>       jz     short UPD_90        ; EXIT WITH CY= 0 IF CAN READ CLOCK NOW
3374 00006786 FB                  <1>       sti                        ; ALLOW INTERRUPTS WHILE WAITING
3375 00006787 E2F1                <1>       loop   UPD_10              ; LOOP TILL READY OR TIMEOUT
3376 00006789 31C0                <1>       xor    eax, eax ; xor ax, ax ; CLEAR RESULTS IF ERROR
3377 0000678B F9                  <1>       stc                        ; SET CARRY FOR ERROR
3378                              <1> UPD_90:
3379 0000678C 59                  <1>       pop    ecx                 ; RESTORE CALLERS REGISTER
3380 0000678D FA                  <1>       cli                        ; INTERRUPTS OFF DURING SET
3381 0000678E C3                  <1>       retn                       ; RETURN WITH CY FLAG SET
3382                              <1>
3383                              <1>       ; 18/04/2021
3384                              <1> RTC_STA:                 ; INITIALIZE REAL TIME CLOCK
3385                              <1>       ;mov   al, CMOS_REG_A      ; ADDRESS REGISTER A AND LOAD DATA MASK
3386                              <1>       ;mov   ah, 26h
3387 0000678F 66B80A26            <1>       mov    ax, (26h*100h)+CMOS_REG_A
3388 00006793 E829000000          <1>       call   CMOS_WRITE          ; INITIALIZE STATUS REGISTER A
3389                              <1>       ;mov   al, CMOS_REG_B      ; SET "SET BIT" FOR CLOCK INITIALIZATION
3390                              <1>       ;mov   ah, 82h
3391 00006798 66B80B82            <1>       mov    ax, (82h*100h)+CMOS_REG_B
3392 0000679C E820000000          <1>       call   CMOS_WRITE          ; AND 24 HOUR MODE TO REGISTER B
3393 000067A1 B00C                <1>       mov    al, CMOS_REG_C      ; ADDRESS REGISTER C
3394 000067A3 E802000000          <1>       call   CMOS_READ           ; READ REGISTER C TO INITIALIZE
3395 000067A8 B00D                <1>       mov    al, CMOS_REG_D      ; ADDRESS REGISTER D
3396                              <1>       ; 18/04/2021
3397                              <1>       ;call  CMOS_READ           ; READ REGISTER D TO INITIALIZE
3398                              <1>       ;retn
3399                              <1>       ;jmp   short CMOS_READ ; 18/04/2021
3400                              <1>
```

```
                        <1>       ; 18/04/2021 - TRDOS 386 v2.0.4
                        <1>       ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
                        <1>       ; 22/08/2014 (Retro UNIX 386 v1)
                        <1>       ; IBM PC/AT BIOS source code ----- 10/06/85 (test4.asm)
                        <1>
                        <1> ;--- CMOS_READ ----------------------------------------------------------------
                        <1> ;           READ BYTE FROM CMOS_SYSTEM CLOCK CONFIGURATION TABLE             :
                        <1> ;                                                                            :
                        <1> ; INPUT: (AL)=     CMOS_TABLE ADDRESS TO BE READ                             :
                        <1> ;           BIT    7 = 0 FOR NMI ENABLED AND 1 FOR NMI DISABLED ON EXIT      :
                        <1> ;           BITS 6-0 = ADDRESS OF TABLE LOCATION TO READ                     :
                        <1> ;                                                                            :
                        <1> ; OUTPUT: (AL) VALUE AT LOCATION (AL) MOVED INTO (AL). IF BIT 7 OF (AL) WAS  :
                        <1> ;           ON THEN NMI LEFT DISABLED, DURING THE CMOS READ BOTH NMI AND     :
                        <1> ;           NORMAL INTERRUPTS ARE DISABLED TO PROTECT CMOS DATA INTEGRITY.   :
                        <1> ;           THE CMOS ADDRESS REGISTER IS POINTED TO A DEFAULT VALUE AND       :
                        <1> ;           THE INTERRUPT FLAG RESTORED TO THE ENTRY STATE ON RETURN.         :
                        <1> ;           ONLY THE (AL) REGISTER AND THE NMI STATE IS CHANGED.              :
                        <1> ;-------------------------------------------------------------------------------
                        <1>
                        <1> CMOS_READ:
000067AA 9C             <1>       pushf                   ; SAVE INTERRUPT ENABLE STATUS AND FLAGS
000067AB D0C0           <1>       rol   al, 1             ; MOVE NMI BIT TO LOW POSITION
000067AD F9             <1>       stc                     ; FORCE NMI BIT ON IN CARRY FLAG
000067AE D0D8           <1>       rcr   al, 1             ; HIGH BIT ON TO DISABLE NMI - OLD IN CY
000067B0 FA             <1>       cli                     ; DISABLE INTERRUPTS
000067B1 E670           <1>       out   CMOS_PORT, al     ; ADDRESS LOCATION AND DISABLE NMI
                        <1>       ; 29/05/2016
                        <1>       ;nop                    ; I/O DELAY
000067B3 E6EB           <1>       out   0EBh, al ; NEWIODELAY ; AWARD BIOS 1999, ATIME.ASM
                        <1>       ;
000067B5 E471           <1>       in    al, CMOS_DATA     ; READ THE REQUESTED CMOS LOCATION
                        <1>       ;push  ax               ; SAVE (AH) REGISTER VALUE AND CMOS BYTE
                        <1>       ; 18/04/2021
000067B7 50             <1>       push  eax
                        <1>       ; 15/03/2015 ; IBM PC/XT Model 286 BIOS source code
                        <1>                    ; ----- 10/06/85 (test4.asm)
000067B8 B01E           <1>       mov   al, CMOS_SHUT_DOWN*2 ; GET ADDRESS OF DEFAULT LOCATION
                        <1>       ;mov  al, CMOS_REG_D*2  ; GET ADDRESS OF DEFAULT LOCATION
000067BA D0D8           <1>       rcr   al, 1             ; PUT ORIGINAL NMI MASK BIT INTO ADDRESS
000067BC E670           <1>       out   CMOS_PORT, al     ; SET DEFAULT TO READ ONLY REGISTER
                        <1>       ;pop  ax                ; RESTORE (AH) AND (AL), CMOS BYTE
                        <1>       ; 18/04/2021
000067BE 58             <1>       pop   eax
000067BF 9D             <1>       popf
000067C0 C3             <1>       retn                    ; RETURN WITH FLAGS RESTORED
                        <1>
                        <1> ; 18/04/2021 (TRDOS 386 v2.0.4)
                        <1> ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>
                        <1> ;--- CMOS_WRITE ---------------------------------------------------------------
                        <1> ;     WRITE BYTE TO CMOS SYSTEM CLOCK CONFIGURATION TABLE                    :
                        <1> ;                                                                            :
                        <1> ; INPUT: (AL)=     CMOS TABLE ADDRESS TO BE WRITTEN TO                       :
                        <1> ;           BIT    7 = 0 FOR NMI ENABLED AND 1 FOR NMI DISABLED ON EXIT      :
                        <1> ;           BITS 6-0 = ADDRESS OF TABLE LOCATION TO WRITE                    :
                        <1> ;      (AH)= NEW VALUE TO BE PLACED IN THE ADDRESSED TABLE LOCATION           :
                        <1> ;                                                                            :
                        <1> ; OUTPUT:   VALUE IN (AH) PLACED IN LOCATION (AL) WITH NMI LEFT DISABLED     :
                        <1> ;           IF BIT 7 OF (AL) IS ON, DURING THE CMOS UPDATE BOTH NMI AND      :
                        <1> ;           NORMAL INTERRUPTS ARE DISABLED TO PROTECT CMOS DATA INTEGRITY.   :
                        <1> ;           THE CMOS ADDRESS REGISTER IS POINTED TO A DEFAULT VALUE AND       :
                        <1> ;           THE INTERRUPT FLAG RESTORED TO THE ENTRY STATE ON RETURN.         :
                        <1> ;           ONLY THE CMOS LOCATION AND THE NMI STATE IS CHANGED.              :
                        <1> ;-------------------------------------------------------------------------------
                        <1>
                        <1> CMOS_WRITE:                   ; WRITE (AH) TO LOCATION (AL)
000067C1 9C             <1>       pushf                   ; SAVE INTERRUPT ENABLE STATUS AND FLAGS
                        <1>       ;push  ax               ; SAVE WORK REGISTER VALUES
                        <1>       ; 18/04/2021
000067C2 50             <1>       push  eax
000067C3 D0C0           <1>       rol   al, 1             ; MOVE NMI BIT TO LOW POSITION
000067C5 F9             <1>       stc                     ; FORCE NMI BIT ON IN CARRY FLAG
000067C6 D0D8           <1>       rcr   al, 1             ; HIGH BIT ON TO DISABLE NMI - OLD IN CY
000067C8 FA             <1>       cli                     ; DISABLE INTERRUPTS
000067C9 E670           <1>       out   CMOS_PORT, al     ; ADDRESS LOCATION AND DISABLE NMI
000067CB 88E0           <1>       mov   al, ah            ; GET THE DATA BYTE TO WRITE
000067CD E671           <1>       out   CMOS_DATA, al     ; PLACE IN REQUESTED CMOS LOCATION
000067CF B01E           <1>       mov   al, CMOS_SHUT_DOWN*2 ; GET ADDRESS OF DEFAULT LOCATION
                        <1>       ;mov  al, CMOS_REG_D*2  ; GET ADDRESS OF DEFAULT LOCATION
000067D1 D0D8           <1>       rcr   al, 1             ; PUT ORIGINAL NMI MASK BIT INTO ADDRESS
000067D3 E670           <1>       out   CMOS_PORT, al     ; SET DEFAULT TO READ ONLY REGISTER
000067D5 90             <1>       nop                     ; I/O DELAY
000067D6 E471           <1>       in    al, CMOS_DATA     ; OPEN STANDBY LATCH
                        <1>       ;pop  ax                ; RESTORE WORK REGISTERS
                        <1>       ; 18/04/2021
000067D8 58             <1>       pop   eax
000067D9 9D             <1>       popf
000067DA C3             <1>       retn
                        <1>
                        <1> ; /// End Of TIMER FUNCTIONS ///
                        
000067DB 90<rep 5h>             Align 16
                        
                                gdt:  ; Global Descriptor Table
                                      ; 02/12/2020
                                      ; (30/07/2015, conforming cs)
                                      ; (26/03/2015)
                                      ; (24/03/2015, tss)
                                      ; (19/03/2015)
                                      ; (29/12/2013)
                                      ;
000067E0 0000000000000000             dw 0, 0, 0, 0       ; NULL descriptor
                                gdt_kcode:
                                      ; 18/08/2014
```

```
                                        ; 8h kernel code segment, base = 00000000h
                                ;dw 0FFFFh, 0, 9E00h, 00CFh      ; KCODE  ; 30/12/2016
000067E8 FFFF0000009ACF00               dw 0FFFFh, 0, 9A00h, 00CFh; KCODE
                                gdt_kdata:
                                        ; 10h kernel data segment, base = 00000000h
000067F0 FFFF00000092CF00               dw 0FFFFh, 0, 9200h, 00CFh; KDATA
                                gdt_ucode:
                                        ; 1Bh user code segment, base address = 400000h ; CORE
                                ;dw 0FBFFh, 0, 0FE40h, 00CFh     ; UCODE  ; 30/12/2016
000067F8 FFFB000040FACF00               dw 0FBFFh, 0, 0FA40h, 00CFh      ; UCODE
                                gdt_udata:
                                        ; 23h user data segment, base address = 400000h ; CORE
00006800 FFFB000040F2CF00               dw 0FBFFh, 0, 0F240h, 00CFh      ; UDATA
                                gdt_tss:
                                        ; Task State Segment
00006808 6700                           dw 0067h ; Limit = 103 ; (104-1, tss size = 104 byte,
                                                         ;  no IO permission in ring 3)
                                gdt_tss0:
0000680A 0000                           dw 0  ; TSS base address, bits 0-15
                                gdt_tss1:
0000680C 00                             db 0  ; TSS base address, bits 16-23
                                        ; 49h
0000680D E9                             db 11101001b ; 0E9h => P=1/DPL=11/0/1/0/B/1 --> B = Task is busy (1)
0000680E 00                             db 0 ; G/0/0/AVL/LIMIT=0000 ; (Limit bits 16-19 = 0000) (G=0, 1 byte)
                                gdt_tss2:
0000680F 00                             db 0  ; TSS base address, bits 24-31

                                        ; 30/11/2020
                                        ; 29/11/2020 - TRDOS v2.0.3
                                        ; VESA VBE3 VIDE BIOS 32 BIT PMI SEGMENTS (16 bit segments)
                                        ; 30h ; VBE3CS
                                _vbe3_CS:  ; vesa vbe3 bios uses this as code seg (same addr with _vbe3_DS)
                                        ; limit = 65536, base addr = 0, P/DPL/1/Type/C/R/A = 9Ah, 16 bit
00006810 FFFF0000009A0000               dw 0FFFFh, 0, 9A00h, 0 ; Note: base addr will be initialized
                                                ; 38h ; VBE3BDS
                                _vbe3_BDS: ; vesa vbe3 bios uses this as equivalent of rombios data segment
                                        ; limit = 1536, base addr = 0, P/DPL/1/Type/E/W/A = 92h, 16 bit
00006818 FF05000000920000               dw 05FFh, 0, 9200h, 0 ; Note: base addr will be initialized
                                                ; 40h ; VBE3A000
                                _A0000Sel: ; VGA default video memory address
                                        ; limit = 65536, base addr = 0A0000h, 16 bit
00006820 FFFF00000A920000               dw 0FFFFh, 0, 920Ah, 0
                                                ; 48h ; VBE3B000
                                _B0000Sel: ; MDA (monochrome) video memory address
                                        ; limit = 65536, base addr = 0B0000h, 16 bit
00006828 FFFF00000B920000               dw 0FFFFh, 0, 920Bh, 0
                                                ; 50h ; VBE3B800
                                _B8000Sel: ; CGA video memory address
                                        ; limit = 32768, base addr = 0B8000h, 16 bit
00006830 FF7F00800B920000               dw 07FFFh, 8000h, 920Bh, 0
                                                ; 58h ; VBE3DS
                                _vbe3_DS: ; vesa vbe3 bios uses this as data seg (CodeSegSel in PMInfoBlock)
                                        ; limit = 65536, base addr = 0, P/DPL/1/Type/E/W/A = 92h, 16 bit
00006838 FFFF000000920000               dw 0FFFFh, 0, 9200h, 0 ; Note: base addr will be initialized
                                                ; 60h ; VBE3SS
                                _vbe3_SS: ; kernel's stack segment but 16 bit version (same stack addr)
                                        ; limit = 1024, base addr = 0, P/DPL/1/Type/E/W/A = 92h, 16 bit
00006840 FF03000000920000               dw 03FFh, 0, 9200h, 0 ; Note: base addr will be initialized
                                                ; 68h ; VBE3ES
                                _vbe3_ES: ; extra 16 bit segment points to buffers in kernel's mem space
                                        ; limit = 2048, base addr = 0, P/DPL/1/Type/E/W/A = 92h, 16 bit
00006848 FF07000000920000               dw 07FFh, 0, 9200h,0 ; Note: base addr will be initialized
                                                ; 70h ; KODE16
                                _16bit_CS: ; 16 bit code segment points to kernel's far return addr
                                        ; limit = 16M, base addr = 0, P/DPL/1/Type/E/W/A = 92h, 16 bit
00006850 FFFF0000009AFF00               dw 0FFFFh, 0, 9A00h, 00FFh ; Note: base addr will be initialized

                                gdt_end:
                                        ;; 9Eh = 1001 1110b (GDT byte 5) P=1/DPL=00/1/TYPE=1110,
                                                                ;; Type= 1 (code)/C=1/R=1/A=0
                                                ; P= Present, DPL=0=ring 0,  1= user (0= system)
                                                ; 1= Code C= Conforming, R= Readable, A = Accessed

                                ;; 9Ah = 1001 1010b (GDT byte 5) P=1/DPL=00/1/TYPE=1010,
                                                                ;; Type= 1 (code)/C=0/R=1/A=0
                                        ; P= Present, DPL=0=ring 0,  1= user (0= system)
                                        ; 1= Code C= non-Conforming, R= Readable, A = Accessed

                                ;; 92h = 1001 0010b (GDT byte 5) P=1/DPL=00/1/TYPE=1010,
                                                                ;; Type= 0 (data)/E=0/W=1/A=0
                                        ; P= Present, DPL=0=ring 0,  1= user (0= system)
                                        ; 0= Data E= Expansion direction (1= down, 0= up)
                                        ; W= Writeable, A= Accessed

                                ;; FEh = 1111 1110b (GDT byte 5) P=1/DPL=11/1/TYPE=1110,
                                                                ;; Type= 1 (code)/C=1/R=1/A=0
                                        ; P= Present, DPL=3=ring 3,  1= user (0= system)
                                        ; 1= Code C= Conforming, R= Readable, A = Accessed

                                ;; FAh = 1111 1010b (GDT byte 5) P=1/DPL=11/1/TYPE=1010,
                                                                ;; Type= 1 (code)/C=0/R=1/A=0
                                        ; P= Present, DPL=3=ring 3,  1= user (0= system)
                                        ; 1= Code C= non-Conforming, R= Readable, A = Accessed

                                ;; F2h = 1111 0010b (GDT byte 5) P=1/DPL=11/1/TYPE=0010,
                                                                ;; Type= 0 (data)/E=0/W=1/A=0
                                        ; P= Present, DPL=3=ring 3,  1= user (0= system)
                                        ; 0= Data E= Expansion direction (1= down, 0= up)

                                ;; CFh = 1100 1111b (GDT byte 6) G=1/B=1/0/AVL=0, Limit=1111b (3)

                                        ;; Limit = FFFFFh (=> FFFFFh+1= 100000h) // bits 0-15, 48-51 //
                                        ;       = 100000h * 1000h (G=1) = 4GB
                                        ;; Limit = FFBFFh (=> FFBFFh+1= FFC00h) // bits 0-15, 48-51 //
                                        ;       = FFC00h * 1000h (G=1) = 4GB - 4MB
```

```
                                        ; G= Granularity (1= 4KB), B= Big (32 bit),
                                        ; AVL= Available to programmers

                                gdtd:
00006858 7700                           dw gdt_end - gdt - 1    ; Limit (size)
0000685A [E0670000]                     dd gdt                  ; Address of the GDT

                                      ; 20/08/2014
                                idtd:
0000685E 7F02                           dw idt_end - idt - 1    ; Limit (size)
00006860 [C07D0100]                     dd idt                  ; Address of the IDT

                                ; 20/02/2017
                                ;;; 11/03/2015
                                %include 'diskdata.s'     ; DISK (BIOS) DATA (initialized)
                              <1> ; ****************************************************************************
                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - diskdata.s
                              <1> ; ----------------------------------------------------------------------------
                              <1> ; Last Update: 24/01/2016 (11/04/2021)
                              <1> ; ----------------------------------------------------------------------------
                              <1> ; Beginning: 24/01/2016
                              <1> ; ----------------------------------------------------------------------------
                              <1> ; Assembler: NASM version 2.11 (trdos386.s)
                              <1> ; ----------------------------------------------------------------------------
                              <1> ; Turkish Rational DOS
                              <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
                              <1> ;
                              <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
                              <1> ; diskdata.inc (11/03/2015)
                              <1> ;
                              <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
                              <1> ; ****************************************************************************
                              <1>
                              <1> ; Retro UNIX 386 v1 Kernel - DISKDATA.INC
                              <1> ; Last Modification: 11/03/2015
                              <1> ;     (Initialized Disk Parameters Data section for 'DISKIO.INC')
                              <1> ;
                              <1>
                              <1> ;---------------------------------------
                              <1> ;     80286 INTERRUPT LOCATIONS :
                              <1> ;     REFERENCED BY POST & BIOS :
                              <1> ;---------------------------------------
                              <1>
00006864 [C7680000]           <1> DISK_POINTER:      dd     MD_TBL6             ; Pointer to Diskette Parameter Table
                              <1>
                              <1> ; IBM PC-XT Model 286 source code ORGS.ASM (06/10/85) - 14/12/2014
                              <1> ;-----------------------------------------------------------------
                              <1> ; DISK_BASE                                     :
                              <1> ;     THIS IS THE SET OF PARAMETERS REQUIRED FOR          :
                              <1> ;     DISKETTE OPERATION. THEY ARE POINTED AT BY THE             :
                              <1> ;     DATA VARIABLE @DISK_POINTER. TO MODIFY THE PARAMETERS,     :
                              <1> ;     BUILD ANOTHER PARAMETER BLOCK AND POINT AT IT       :
                              <1> ;-----------------------------------------------------------------
                              <1>
                              <1> ;DISK_BASE:
                              <1> ;     DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
                              <1> ;     DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
                              <1> ;     DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
                              <1> ;     DB     2            ; 512 BYTES/SECTOR
                              <1> ;     ;DB    15           ; EOT (LAST SECTOR ON TRACK)
                              <1> ;     db     18           ; (EOT for 1.44MB diskette)
                              <1> ;     DB     01BH         ; GAP LENGTH
                              <1> ;     DB     0FFH         ; DTL
                              <1> ;     ;DB    054H         ; GAP LENGTH FOR FORMAT
                              <1> ;     db     06ch         ; (for 1.44MB dsikette)
                              <1> ;     DB     0F6H         ; FILL BYTE FOR FORMAT
                              <1> ;     DB     15           ; HEAD SETTLE TIME (MILLISECONDS)
                              <1> ;     DB     8            ; MOTOR START TIME (1/8 SECONDS)
                              <1>
                              <1> ;---------------------------------------
                              <1> ;     ROM BIOS DATA AREAS       :
                              <1> ;---------------------------------------
                              <1>
                              <1> ;DATA SEGMENT AT 40H            ; ADDRESS= 0040:0000
                              <1>
                              <1> ;@EQUIP_FLAG DW     ?           ; INSTALLED HARDWARE FLAGS
                              <1>
                              <1> ;---------------------------------------
                              <1> ;     DISKETTE DATA AREAS       :
                              <1> ;---------------------------------------
                              <1>
                              <1> ;@SEEK_STATUS      DB     ?     ; DRIVE RECALIBRATION STATUS
                              <1> ;                                ; BIT 3-0 = DRIVE 3-0 RECALIBRATION
                              <1> ;                                ; BEFORE NEXT SEEK IF BIT IS = 0
                              <1> ;@MOTOR_STATUS     DB     ?     ; MOTOR STATUS
                              <1> ;                                ; BIT 3-0 = DRIVE 3-0 CURRENTLY RUNNING
                              <1> ;                                ; BIT 7 = CURRENT OPERATION IS A WRITE
                              <1> ;@MOTOR_COUNT      DB     ?     ; TIME OUT COUNTER FOR MOTOR(S) TURN OFF
                              <1> ;@DSKETTE_STATUS DB       ?     ; RETURN CODE STATUS BYTE
                              <1> ;                                ; CMD_BLOCK  IN STACK FOR DISK OPERATION
                              <1> ;@NEC_STATUS DB     7 DUP(?)    ; STATUS BYTES FROM DISKETTE OPERATION
                              <1>
                              <1> ;---------------------------------------
                              <1> ;     POST AND BIOS WORK DATA AREA     :
                              <1> ;---------------------------------------
                              <1>
                              <1> ;@INTR_FLAG  DB     ?           ; FLAG INDICATING AN INTERRUPT HAPPENED
                              <1>
                              <1> ;---------------------------------------
                              <1> ;     TIMER DATA AREA           :
                              <1> ;---------------------------------------
                              <1>
                              <1> ; 17/12/2014  (IRQ 0 - INT 08H)
                              <1> ;TIMER_LOW  equ    46Ch         ; Timer ticks (counter)  @ 40h:006Ch
                              <1> ;TIMER_HIGH equ    46Eh         ; (18.2 timer ticks per second)
```

```
                        <1> ;TIMER_OFL   equ    470h         ; Timer - 24 hours flag  @ 40h:0070h
                        <1>
                        <1> ;----------------------------------------
                        <1> ;     ADDITIONAL MEDIA DATA            :
                        <1> ;----------------------------------------
                        <1>
                        <1> ;@LASTRATE   DB     ?            ; LAST DISKETTE DATA RATE SELECTED
                        <1> ;@DSK_STATE  DB     ?            ; DRIVE 0 MEDIA STATE
                        <1> ;            DB     ?            ; DRIVE 1 MEDIA STATE
                        <1> ;            DB     ?            ; DRIVE 0 OPERATION START STATE
                        <1> ;            DB     ?            ; DRIVE 1 OPERATION START STATE
                        <1> ;@DSK_TRK    DB     ?            ; DRIVE 0 PRESENT CYLINDER
                        <1> ;            DB     ?            ; DRIVE 1 PRESENT CYLINDER
                        <1>
                        <1> ;DATA        ENDS                ; END OF BIOS DATA SEGMENT
                        <1>
                        <1> ;--------------------------------------------------------
                        <1> ;     DRIVE TYPE TABLE                          :
                        <1> ;--------------------------------------------------------
                        <1>              ; 16/02/2015 (unix386.s, 32 bit modifications)
                        <1> DR_TYPE:
00006868 01             <1>              DB     01           ; DRIVE TYPE, MEDIA TABLE
                        <1>               ;DW   MD_TBL1
00006869 [86680000]     <1>              dd     MD_TBL1
0000686D 82             <1>              DB     02+BIT7ON
                        <1>              ;DW    MD_TBL2
0000686E [93680000]     <1>             dd     MD_TBL2
00006872 02             <1> DR_DEFAULT:  DB     02
                        <1>              ;DW    MD_TBL3
00006873 [A0680000]     <1>              dd    MD_TBL3
00006877 03             <1>              DB     03
                        <1>              ;DW    MD_TBL4
00006878 [AD680000]     <1>              dd     MD_TBL4
0000687C 84             <1>              DB     04+BIT7ON
                        <1>              ;DW    MD_TBL5
0000687D [BA680000]     <1>              dd     MD_TBL5
00006881 04             <1>              DB     04
                        <1>              ;DW    MD_TBL6
00006882 [C7680000]     <1>              dd     MD_TBL6
                        <1> DR_TYPE_E    equ $   ; END OF TABLE
                        <1> ;DR_CNT      EQU    (DR_TYPE_E-DR_TYPE)/3
                        <1> DR_CNT       equ    (DR_TYPE_E-DR_TYPE)/5
                        <1> ;--------------------------------------------------------
                        <1> ;     MEDIA/DRIVE PARAMETER TABLES                 :
                        <1> ;--------------------------------------------------------
                        <1> ;--------------------------------------------------------
                        <1> ;     360 KB MEDIA IN 360 KB DRIVE                 :
                        <1> ;--------------------------------------------------------
                        <1> MD_TBL1:
00006886 DF             <1>       DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
00006887 02             <1>       DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
00006888 25             <1>       DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
00006889 02             <1>       DB     2            ; 512 BYTES/SECTOR
0000688A 09             <1>       DB     09           ; EOT (LAST SECTOR ON TRACK)
0000688B 2A             <1>       DB     02AH         ; GAP LENGTH
0000688C FF             <1>       DB     0FFH         ; DTL
0000688D 50             <1>       DB     050H         ; GAP LENGTH FOR FORMAT
0000688E F6             <1>       DB     0F6H         ; FILL BYTE FOR FORMAT
0000688F 0F             <1>       DB     15           ; HEAD SETTLE TIME (MILLISECONDS)
00006890 08             <1>       DB     8            ; MOTOR START TIME (1/8 SECONDS)
00006891 27             <1>       DB     39           ; MAX. TRACK NUMBER
00006892 80             <1>       DB     RATE_250     ; DATA TRANSFER RATE
                        <1> ;--------------------------------------------------------
                        <1> ;     360 KB MEDIA IN 1.2 MB DRIVE                 :
                        <1> ;--------------------------------------------------------
                        <1> MD_TBL2:
00006893 DF             <1>       DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
00006894 02             <1>       DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
00006895 25             <1>       DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
00006896 02             <1>       DB     2            ; 512 BYTES/SECTOR
00006897 09             <1>       DB     09           ; EOT (LAST SECTOR ON TRACK)
00006898 2A             <1>       DB     02AH         ; GAP LENGTH
00006899 FF             <1>       DB     0FFH         ; DTL
0000689A 50             <1>       DB     050H         ; GAP LENGTH FOR FORMAT
0000689B F6             <1>       DB     0F6H         ; FILL BYTE FOR FORMAT
0000689C 0F             <1>       DB     15           ; HEAD SETTLE TIME (MILLISECONDS)
0000689D 08             <1>       DB     8            ; MOTOR START TIME (1/8 SECONDS)
0000689E 27             <1>       DB     39           ; MAX. TRACK NUMBER
0000689F 40             <1>       DB     RATE_300     ; DATA TRANSFER RATE
                        <1> ;--------------------------------------------------------
                        <1> ;     1.2 MB MEDIA IN 1.2 MB DRIVE                 :
                        <1> ;--------------------------------------------------------
                        <1> MD_TBL3:
000068A0 DF             <1>       DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
000068A1 02             <1>       DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
000068A2 25             <1>       DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
000068A3 02             <1>       DB     2            ; 512 BYTES/SECTOR
000068A4 0F             <1>       DB     15           ; EOT (LAST SECTOR ON TRACK)
000068A5 1B             <1>       DB     01BH         ; GAP LENGTH
000068A6 FF             <1>       DB     0FFH         ; DTL
000068A7 54             <1>       DB     054H         ; GAP LENGTH FOR FORMAT
000068A8 F6             <1>       DB     0F6H         ; FILL BYTE FOR FORMAT
000068A9 0F             <1>       DB     15           ; HEAD SETTLE TIME (MILLISECONDS)
000068AA 08             <1>       DB     8            ; MOTOR START TIME (1/8 SECONDS)
000068AB 4F             <1>       DB     79           ; MAX. TRACK NUMBER
000068AC 00             <1>       DB     RATE_500     ; DATA TRANSFER RATE
                        <1> ;--------------------------------------------------------
                        <1> ;     720 KB MEDIA IN 720 KB DRIVE                 :
                        <1> ;--------------------------------------------------------
                        <1> MD_TBL4:
000068AD DF             <1>       DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
000068AE 02             <1>       DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
000068AF 25             <1>       DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
000068B0 02             <1>       DB     2            ; 512 BYTES/SECTOR
000068B1 09             <1>       DB     09           ; EOT (LAST SECTOR ON TRACK)
```

```
3245 000068B2 2A                <1>       DB     02AH          ; GAP LENGTH
3246 000068B3 FF                <1>       DB     0FFH          ; DTL
3247 000068B4 50                <1>       DB     050H          ; GAP LENGTH FOR FORMAT
3248 000068B5 F6                <1>       DB     0F6H          ; FILL BYTE FOR FORMAT
3249 000068B6 0F                <1>       DB     15            ; HEAD SETTLE TIME (MILLISECONDS)
3250 000068B7 08                <1>       DB     8             ; MOTOR START TIME (1/8 SECONDS)
3251 000068B8 4F                <1>       DB     79            ; MAX. TRACK NUMBER
3252 000068B9 80                <1>       DB     RATE_250      ; DATA TRANSFER RATE
3253                            <1> ;--------------------------------------------------------
3254                            <1> ;     720 KB MEDIA IN 1.44 MB DRIVE                :
3255                            <1> ;--------------------------------------------------------
3256                            <1> MD_TBL5:
3257 000068BA DF                <1>       DB     11011111B     ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
3258 000068BB 02                <1>       DB     2             ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
3259 000068BC 25                <1>       DB     MOTOR_WAIT    ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
3260 000068BD 02                <1>       DB     2             ; 512 BYTES/SECTOR
3261 000068BE 09                <1>       DB     09            ; EOT (LAST SECTOR ON TRACK)
3262 000068BF 2A                <1>       DB     02AH          ; GAP LENGTH
3263 000068C0 FF                <1>       DB     0FFH          ; DTL
3264 000068C1 50                <1>       DB     050H          ; GAP LENGTH FOR FORMAT
3265 000068C2 F6                <1>       DB     0F6H          ; FILL BYTE FOR FORMAT
3266 000068C3 0F                <1>       DB     15            ; HEAD SETTLE TIME (MILLISECONDS)
3267 000068C4 08                <1>       DB     8             ; MOTOR START TIME (1/8 SECONDS)
3268 000068C5 4F                <1>       DB     79            ; MAX. TRACK NUMBER
3269 000068C6 80                <1>       DB     RATE_250      ; DATA TRANSFER RATE
3270                            <1> ;--------------------------------------------------------
3271                            <1> ;     1.44 MB MEDIA IN 1.44 MB DRIVE               :
3272                            <1> ;--------------------------------------------------------
3273                            <1> MD_TBL6:
3274 000068C7 AF                <1>       DB     10101111B     ; SRT=A, HD UNLOAD=0F - 1ST SPECIFY BYTE
3275 000068C8 02                <1>       DB     2             ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
3276 000068C9 25                <1>       DB     MOTOR_WAIT    ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
3277 000068CA 02                <1>       DB     2             ; 512 BYTES/SECTOR
3278 000068CB 12                <1>       DB     18            ; EOT (LAST SECTOR ON TRACK)
3279 000068CC 1B                <1>       DB     01BH          ; GAP LENGTH
3280 000068CD FF                <1>       DB     0FFH          ; DTL
3281 000068CE 6C                <1>       DB     06CH          ; GAP LENGTH FOR FORMAT
3282 000068CF F6                <1>       DB     0F6H          ; FILL BYTE FOR FORMAT
3283 000068D0 0F                <1>       DB     15            ; HEAD SETTLE TIME (MILLISECONDS)
3284 000068D1 08                <1>       DB     8             ; MOTOR START TIME (1/8 SECONDS)
3285 000068D2 4F                <1>       DB     79            ; MAX. TRACK NUMBER
3286 000068D3 00                <1>       DB     RATE_500      ; DATA TRANSFER RATE
3287                            <1>
3288                            <1>
3289                            <1> ; << diskette.inc >>
3290                            <1> ; ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
3291                            <1> ;
3292                            <1> ;----------------------------------------
3293                            <1> ;     ROM BIOS DATA AREAS       :
3294                            <1> ;----------------------------------------
3295                            <1>
3296                            <1> ;DATA SEGMENT AT 40H            ; ADDRESS= 0040:0000
3297                            <1>
3298                            <1> ;----------------------------------------
3299                            <1> ;     FIXED DISK DATA AREAS           :
3300                            <1> ;----------------------------------------
3301                            <1>
3302                            <1> ;DISK_STATUS1:     DB     0      ; FIXED DISK STATUS
3303                            <1> ;HF_NUM:           DB     0      ; COUNT OF FIXED DISK DRIVES
3304                            <1> ;CONTROL_BYTE:     DB     0      ; HEAD CONTROL BYTE
3305                            <1> ;@PORT_OFF   DB    ?             ; RESERVED (PORT OFFSET)
3306                            <1>
3307                            <1> ;----------------------------------------
3308                            <1> ;     ADDITIONAL MEDIA DATA           :
3309                            <1> ;----------------------------------------
3310                            <1>
3311                            <1> ;@LASTRATE   DB     ?             ; LAST DISKETTE DATA RATE SELECTED
3312                            <1> ;HF_STATUS   DB     0             ; STATUS REGISTER
3313                            <1> ;HF_ERROR    DB     0             ; ERROR REGISTER
3314                            <1> ;HF_INT_FLAG DB     0             ; FIXED DISK INTERRUPT FLAG
3315                            <1> ;HF_CNTRL    DB     0             ; COMBO FIXED DISK/DISKETTE CARD BIT 0=1
3316                            <1> ;@DSK_STATE  DB     ?             ; DRIVE 0 MEDIA STATE
3317                            <1> ;            DB     ?             ; DRIVE 1 MEDIA STATE
3318                            <1> ;            DB     ?             ; DRIVE 0 OPERATION START STATE
3319                            <1> ;            DB     ?             ; DRIVE 1 OPERATION START STATE
3320                            <1> ;@DSK_TRK    DB     ?             ; DRIVE 0 PRESENT CYLINDER
3321                            <1> ;            DB     ?             ; DRIVE 1 PRESENT CYLINDER
3322                            <1>
3323                            <1> ;DATA        ENDS                 ; END OF BIOS DATA SEGMENT
3324                            <1> ;
3325                            <1> ; ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
3326                            <1>
3327                            <1> ERR_TBL:
3328 000068D4 E0                <1>       db     NO_ERR
3329 000068D5 024001BB          <1>       db     BAD_ADDR_MARK,BAD_SEEK,BAD_CMD,UNDEF_ERR
3330 000068D9 04BB100A          <1>       db     RECORD_NOT_FND,UNDEF_ERR,BAD_ECC,BAD_SECTOR
3331                            <1>
3332                            <1> ; 17/12/2014 (mov ax, [cfd])
3333                            <1> ; 11/12/2014
3334 000068DD 00                <1> cfd:         db     0      ; current floppy drive (for GET_PARM)
3335                            <1> ; 17/12/2014                ; instead of 'DISK_POINTER'
3336 000068DE 01                <1> pfd:         db     1      ; previous floppy drive (for GET_PARM)
3337                            <1>                             ; (initial value of 'pfd
3338                            <1>                             ; must be different then 'cfd' value
3339                            <1>                             ; to force updating/initializing
3340                            <1>                             ; current drive parameters)
3341 000068DF 90                <1> align 2
3342                            <1>
3343 000068E0 F001              <1> HF_PORT:     dw     1F0h  ; Default = 1F0h
3344                            <1>                             ; (170h)
3345 000068E2 F603              <1> HF_REG_PORT: dw     3F6h  ; HF_PORT + 206h
3346                            <1>
3347                            <1> ; 05/01/2015
3348 000068E4 00                <1> hf_m_s:      db     0      ; (0 = Master, 1 = Slave)
3349                            <1>
```

```
                        <1> ; ****************************************************************************
                            Align 2

                            ; 04/11/2014 (Retro UNIX 386 v1)
000068E6 0000               mem_1m_1k:   dw 0  ; Number of contiguous KB between
                                               ; 1 and 16 MB, max. 3C00h = 15 MB.
000068E8 0000               mem_16m_64k: dw 0  ; Number of contiguous 64 KB blocks
                                             ; between 16 MB and 4 GB.

                            ; 12/11/2014 (Retro UNIX 386 v1)
000068EA 00                 boot_drv:    db 0 ; boot drive number (physical)
                            ; 24/11/2014
000068EB 00                 drv:       db 0
000068EC 00                 last_drv:    db 0 ; last hdd
000068ED 00                 hdc:         db 0  ; number of hard disk drives
                                             ; (present/detected)

                            ; 24/11/2014 (Retro UNIX 386 v1)
                            ; Physical drive type & flags
000068EE 00                 fd0_type:    db 0  ; floppy drive type
000068EF 00                 fd1_type:    db 0  ; 4 = 1.44 Mb, 80 track, 3.5" (18 spt)
                                             ; 6 = 2.88 Mb, 80 track, 3.5" (36 spt)
                                             ; 3 = 720 Kb, 80 track, 3.5" (9 spt)
                                             ; 2 = 1.2 Mb, 80 track, 5.25" (15 spt)
                                             ; 1 = 360 Kb, 40 track, 5.25" (9 spt)
000068F0 00                 hd0_type:    db 0  ; EDD status for hd0 (bit 7 = present flag)
000068F1 00                 hd1_type:    db 0  ; EDD status for hd1 (bit 7 = present flag)
000068F2 00                 hd2_type:    db 0  ; EDD status for hd2 (bit 7 = present flag)
000068F3 00                 hd3_type:    db 0  ; EDD status for hd3 (bit 7 = present flag)
                                             ; bit 0 - Fixed disk access subset supported
                                             ; bit 1 - Drive locking and ejecting
                                             ; bit 2 - Enhanced disk drive support
                                             ; bit 3 = Reserved (64 bit EDD support)
                                             ; (If bit 0 is '1' Retro UNIX 386 v1
                                             ; will interpret it as 'LBA ready'!)

                                ; 11/03/2015 - 10/07/2015
000068F4 00000000000000000-     drv.cylinders: dw 0,0,0,0,0,0,0
000068FD 0000000000
00006902 00000000000000000-     drv.heads:     dw 0,0,0,0,0,0,0
0000690B 0000000000
00006910 00000000000000000-     drv.spt:       dw 0,0,0,0,0,0,0
00006919 0000000000
0000691E 00000000000000000-     drv.size:      dd 0,0,0,0,0,0,0
00006927 00000000000000000-
00006930 00000000000000000-
00006939 00
0000693A 00000000000000         drv.status:    db 0,0,0,0,0,0,0
00006941 00000000000000         drv.error:     db 0,0,0,0,0,0,0

                                Align 2

                                ;;; 11/03/2015
                                %include 'kybdata.s'      ; KEYBOARD (BIOS) DATA
                              <1> ; ****************************************************************************
                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - kybdata.s
                              <1> ; ----------------------------------------------------------------------------
                              <1> ; Last Update: 17/01/2016
                              <1> ; ----------------------------------------------------------------------------
                              <1> ; Beginning: 17/01/2016
                              <1> ; ----------------------------------------------------------------------------
                              <1> ; Assembler: NASM version 2.11 (trdos386.s)
                              <1> ; ----------------------------------------------------------------------------
                              <1> ; Turkish Rational DOS
                              <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
                              <1> ;
                              <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
                              <1> ; kybdata.inc (11/03/2015)
                              <1> ;
                              <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
                              <1> ; ****************************************************************************
                              <1>
                              <1> ; Retro UNIX 386 v1 Kernel - KYBDATA.INC
                              <1> ; Last Modification: 11/03/2015
                              <1> ;             (Data Section for 'KEYBOARD.INC')
                              <1> ;
                              <1> ; ///////// KEYBOARD DATA ///////////////
                              <1>
                              <1> ; 05/12/2014
                              <1> ; 04/12/2014 (derived from pc-xt-286 bios source code -1986-)
                              <1> ; 03/06/86  KEYBOARD BIOS
                              <1>
                              <1> ;--------------------------------------------------------------------------------
                              <1> ;     KEY IDENTIFICATION SCAN TABLES
                              <1> ;--------------------------------------------------------------------------------
                              <1>
                              <1> ;-----      TABLES FOR ALT CASE ------------
                              <1> ;-----      ALT-INPUT-TABLE
00006948 524F50514B           <1> K30:  db    82,79,80,81,75
0000694D 4C4D474849           <1>       db    76,77,71,72,73      ; 10 NUMBER ON KEYPAD
                              <1> ;-----      SUPER-SHIFT-TABLE
00006952 101112131415         <1>       db    16,17,18,19,20,21   ; A-Z TYPEWRITER CHARS
00006958 161718191E1F         <1>       db    22,23,24,25,30,31
0000695E 202122232425         <1>       db    32,33,34,35,36,37
00006964 262C2D2E2F30         <1>       db    38,44,45,46,47,48
0000696A 3132                 <1>       db    49,50
                              <1>
                              <1> ;-----      TABLE OF SHIFT KEYS AND MASK VALUES
                              <1> ;-----      KEY_TABLE
0000696C 52                   <1> _K6:  db    INS_KEY             ; INSERT KEY
0000696D 3A4546381D           <1>       db    CAPS_KEY,NUM_KEY,SCROLL_KEY,ALT_KEY,CTL_KEY
00006972 2A36                 <1>       db    LEFT_KEY,RIGHT_KEY
                              <1> _K6L  equ   $-_K6
                              <1>
```

```
                                <1> ;-----      MASK_TABLE
00006974 80                     <1> _K7:  db    INS_SHIFT           ; INSERT MODE SHIFT
00006975 4020100804             <1>       db    CAPS_SHIFT,NUM_SHIFT,SCROLL_SHIFT,ALT_SHIFT,CTL_SHIFT
0000697A 0201                   <1>       db    LEFT_SHIFT,RIGHT_SHIFT
                                <1>
                                <1> ;-----      TABLES FOR CTRL CASE ;---- CHARACTERS ------
0000697C 1BFF00FFFFFF           <1> _K8:  db    27,-1,0,-1,-1,-1    ; Esc, 1, 2, 3, 4, 5
00006982 1EFFFFFFFF1F           <1>       db    30,-1,-1,-1,-1,31   ; 6, 7, 8, 9, 0, -
00006988 FF7FFF111705           <1>       db    -1,127,-1,17,23,5   ; =, Bksp, Tab, Q, W, E
0000698E 12141915090F           <1>       db    18,20,25,21,9,15    ; R, T, Y, U, I, O
00006994 101B1D0AFF01           <1>       db    16,27,29,10,-1,1    ; P, [, ], Enter, Ctrl, A
0000699A 13040607080A           <1>       db    19,4,6,7,8,10       ; S, D, F, G, H, J
000069A0 0B0CFFFFFFFF           <1>       db    11,12,-1,-1,-1,-1   ; K, L, :, ', `, LShift
000069A6 1C1A18031602           <1>       db    28,26,24,3,22,2           ; Bkslash, Z, X, C, V, B
000069AC 0E0DFFFFFFFF           <1>       db    14,13,-1,-1,-1,-1   ; N, M, ,, ., /, RShift
000069B2 96FF20FF               <1>       db    150,-1,' ',-1       ; *, ALT, Spc, CL
                                <1>       ;                         ;----- FUNCTIONS ------
000069B6 5E5F60616263           <1>       db    94,95,96,97,98,99   ; F1 - F6
000069BC 64656667FFFF           <1>       db    100,101,102,103,-1,-1     ; F7 - F10, NL, SL
000069C2 778D848E738F           <1>       db    119,141,132,142,115,143   ; Home, Up, PgUp, -, Left, Pad5
000069C8 749075917692           <1>       db    116,144,117,145,118,146 ; Right, +, End, Down, PgDn, Ins
000069CE 93FFFFFF898A           <1>       db    147,-1,-1,-1,137,138      ; Del, SysReq, Undef, WT, F11, F12
                                <1>
                                <1> ;-----      TABLES FOR LOWER CASE ----------
000069D4 1B3132333435363738-    <1> K10:  db    27,'1234567890-=',8,9
000069DD 39302D3D0809           <1>
000069E3 71776572747975696F-    <1>       db    'qwertyuiop[]',13,-1,'asdfghjkl;',39
000069EC 705B5D0DFF61736466-    <1>
000069F5 67686A6B6C3B27         <1>
000069FC 60FF5C7A786376626E-    <1>       db    96,-1,92,'zxcvbnm,./',-1,'*',-1,' ',-1
00006A05 6D2C2E2FFF2AFF20FF     <1>
                                <1> ;-----      LC TABLE SCAN
00006A0E 3B3C3D3E3F             <1>       db    59,60,61,62,63      ; BASE STATE OF F1 - F10
00006A13 4041424344             <1>       db    64,65,66,67,68
00006A18 FFFF                   <1>       db    -1,-1               ; NL, SL
                                <1>
                                <1> ;-----      KEYPAD TABLE
00006A1A 474849FF4BFF           <1> K15:  db    71,72,73,-1,75,-1   ; BASE STATE OF KEYPAD KEYS
00006A20 4DFF4F50515253         <1>       db    77,-1,79,80,81,82,83
00006A27 FFFF5C8586             <1>       db    -1,-1,92,133,134    ; SysRq, Undef, WT, F11, F12
                                <1>
                                <1> ;-----      TABLES FOR UPPER CASE ----------
00006A2C 1B21402324255E262A-    <1> K11:  db    27,'!@#$%',94,'&*()_+',8,0
00006A35 28295F2B0800           <1>
00006A3B 5157455254595549 4F-   <1>       db    'QWERTYUIOP{}',13,-1,'ASDFGHJKL:"'
00006A44 507B7D0DFF41534446-    <1>
00006A4D 47484A4B4C3A22         <1>
00006A54 7EFF7C5A584356424E-    <1>       db    126,-1,'|ZXCVBNM<>?',-1,'*',-1,' ',-1
00006A5D 4D3C3E3FFF2AFF20FF     <1>
                                <1> ;-----      UC TABLE SCAN
00006A66 5455565758             <1> K12:  db    84,85,86,87,88      ; SHIFTED STATE OF F1 - F10
00006A6B 595A5B5C5D             <1>       db    89,90,91,92,93
00006A70 FFFF                   <1>       db    -1,-1               ; NL, SL
                                <1>
                                <1> ;-----      NUM STATE TABLE
00006A72 3738392D3435362B31-    <1> K14:  db    '789-456+1230.'     ; NUMLOCK STATE OF KEYPAD KEYS
00006A7B 3233302E               <1>
                                <1>       ;
00006A7F FFFF7C8788             <1>       db    -1,-1,124,135,136   ; SysRq, Undef, WT, F11, F12
                                <1>
                                <1> ; 26/08/2014
                                <1> ; Retro UNIX 8086 v1 - UNIX.ASM (03/03/2014)
                                <1> ; Derived from IBM "pc-at"
                                <1> ; rombios source code (06/10/1985)
                                <1> ; 'dseg.inc'
                                <1>
                                <1> ;----------------------------------------
                                <1> ;     SYSTEM DATA AREA                   ;
                                <1> ;----------------------------------------
00006A84 00                     <1> BIOS_BREAK   db     0            ; BIT 7=1 IF BREAK KEY HAS BEEN PRESSED
                                <1>
                                <1> ;----------------------------------------
                                <1> ;     KEYBOARD DATA AREAS              ;
                                <1> ;----------------------------------------
                                <1>
00006A85 00                     <1> KB_FLAG      db     0            ; KEYBOARD SHIFT STATE AND STATUS FLAGS
00006A86 00                     <1> KB_FLAG_1    db     0            ; SECOND BYTE OF KEYBOARD STATUS
00006A87 00                     <1> KB_FLAG_2    db     0            ; KEYBOARD LED FLAGS
00006A88 00                     <1> KB_FLAG_3    db     0            ; KEYBOARD MODE STATE AND TYPE FLAGS
00006A89 00                     <1> ALT_INPUT    db     0            ; STORAGE FOR ALTERNATE KEY PAD ENTRY
00006A8A [9A6A0000]             <1> BUFFER_START dd     KB_BUFFER    ; OFFSET OF KEYBOARD BUFFER START
00006A8E [BA6A0000]             <1> BUFFER_END   dd     KB_BUFFER + 32 ; OFFSET OF END OF BUFFER
00006A92 [9A6A0000]             <1> BUFFER_HEAD  dd     KB_BUFFER    ; POINTER TO HEAD OF KEYBOARD BUFFER
00006A96 [9A6A0000]             <1> BUFFER_TAIL  dd     KB_BUFFER    ; POINTER TO TAIL OF KEYBOARD BUFFER
                                <1> ; ------     HEAD = TAIL  INDICATES THAT THE BUFFER IS EMPTY
00006A9A 0000<rep 10h>          <1> KB_BUFFER    times  16 dw 0       ; ROOM FOR 16 SCAN CODE ENTRIES
                                <1>
                                <1> ; /// End Of KEYBOARD DATA ///
                                    %include 'vidata.s'       ; VIDEO (BIOS) DATA
                                <1> ; ****************************************************************************
                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.3 - vidata.s
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Last Update: 24/11/2020
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Beginning: 16/01/2016
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Assembler: NASM version 2.15 (trdos386.s)
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Turkish Rational DOS
                                <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
                                <1> ;
                                <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
                                <1> ; vidata.inc (11/03/2015)
                                <1> ;
                                <1> ; Derived from 'IBM PC-AT' BIOS source code (1985)
```

```
                        <1> ; ****************************************************************************
                        <1>
                        <1> ; Retro UNIX 386 v1 Kernel - VIDATA.S
                        <1> ; Last Modification: 11/03/2015
                        <1> ;                (Data section for 'VIDEO.INC')
                        <1> ;
                        <1> ; ///////// VIDEO DATA ///////////////
                        <1>
                        <1> ;----------------------------------------
                        <1> ;     VIDEO DISPLAY DATA AREA          ;
                        <1> ;----------------------------------------
00006ABA 03             <1> CRT_MODE:    db     3      ; CURRENT DISPLAY MODE (TYPE)
00006ABB 29             <1> CRT_MODE_SET:      db     29h    ; CURRENT SETTING OF THE 3X8 REGISTER
                        <1>                            ; (29h default setting for video mode 3)
                        <1>                            ; Mode Select register Bits
                        <1>                            ;   BIT 0 - 80x25 (1), 40x25 (0)
                        <1>                            ;   BIT 1 - ALPHA (0), 320x200 GRAPHICS (1)
                        <1>                            ;   BIT 2 - COLOR (0), BW (1)
                        <1>                            ;   BIT 3 - Video Sig. ENABLE (1), DISABLE (0)
                        <1>                            ;   BIT 4 - 640x200 B&W Graphics Mode (1)
                        <1>                            ;   BIT 5 - ALPHA mode BLINKING (1)
                        <1>                            ;   BIT 6, 7 - Not Used
                        <1>
                        <1> ; Mode 0 - 2Ch = 101100b  ; 40x25 text, 16 gray colors
                        <1> ; Mode 1 - 28h = 101000b  ; 40x25 text, 16 fore colors, 8 back colors
                        <1> ; Mode 2 - 2Dh = 101101b  ; 80x25 text, 16 gray colors
                        <1> ; Mode 3 - 29h = 101001b  ; 80x25 text, 16 fore color, 8 back color
                        <1> ; Mode 4 - 2Ah = 101010b  ; 320x200 graphics, 4 colors
                        <1> ; Mode 5 - 2Eh = 101110b  ; 320x200 graphics, 4 gray colors
                        <1> ; Mode 6 - 1Eh = 011110b  ; 640x200 graphics, 2 colors
                        <1> ; Mode 7 - 29h = 101001b  ; 80x25 text, black & white colors
                        <1> ; Mode & 37h = Video signal OFF
                        <1>
                        <1> ; 24/06/2016
00006ABC 50             <1> CRT_COLS:    db     80     ; Number of columns
                        <1>
                        <1> ; 01/07/2016
00006ABD 00             <1> CRT_PALETTE: db     0      ; Current palette setting
                        <1>
                        <1> ; 03/07/2016
00006ABE 10             <1> CHAR_HEIGHT: db     16     ; Default character height
00006ABF 60             <1> VGA_VIDEO_CTL:     db     60h    ; ROM BIOS DATA AREA Offset 87h
00006AC0 F9             <1> VGA_SWITCHES:      db     0F9h   ; Feature Bit Switches (the basic screen)
00006AC1 51             <1> VGA_MODESET_CTL: db       051h   ; Basic mode set options (VGA video flags)
                        <1>                            ; ROM BIOS DATA AREA Offset 89h
                        <1>                            ; Bit 7, 4 : Mode
                        <1>                            ;       01 : 400-line mode
                        <1>                            ; Bit 6 : Display switch enabled = 1
                        <1>                            ; Bit 5 : Reserved = 0
                        <1>                            ; Bit 3 : Default palette loading
                        <1>                            ;       disabled = 0
                        <1>                            ; Bit 2 : Color monitor = 0
                        <1>                            ; Bit 1 = Gray scale summing
                        <1>                            ;       disabled = 0
                        <1>                            ; Bit 0 = VGA active = 1
00006AC2 19             <1> VGA_ROWS:    db     25
                        <1>
                        <1> ; 16/01/2016
                        <1> chr_attrib:  ; Character color/attributes for video pages (0 to 7)
00006AC3 0707070707070707 <1>       db     07h, 07h, 07h, 07h, 07h, 07h, 07h, 07h
                        <1> ; 30/01/2016
                        <1> vmode:
00006ACB 0303030303030303 <1>       db     3,3,3,3,3,3,3,3 ; video modes for pseudo screens
                        <1>
                        <1> CURSOR_MODE: ; cursor start (ch) = 14, cursor end (cl) = 15
00006AD3 0F0E           <1>       db     15, 14 ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
                        <1>
                        <1> ;align 4
                        <1> ;VGA_BASE: ; 26/07/2016
                        <1> ;     dd     0B8000h  ; (Mode < 0Dh) or 0A0000h (mode >= 0Dh)
                        <1>
00006AD5 90             <1> align 2
                        <1>
                        <1> vga_modes:
                        <1>       ; 25/07/2016
                        <1>       ; 09/07/2016
                        <1>       ; 03/07/2016
                        <1>       ; valid (implemented) video modes (>7, extension to IBM PC CGA modes)
00006AD6 0302010007040506 <1>       db     03h, 02h, 01h, 00h, 07h, 04h, 05h, 06h
                        <1> vga_g_modes: ; 31/07/2016
00006ADE 13F0126A0D0E1011 <1>       db     13h, 0F0h, 12h, 6Ah, 0Dh, 0Eh, 10h, 11h
                        <1> vga_mode_count equ $ - vga_modes
                        <1> vga_g_mode_count equ $ - vga_g_modes
                        <1>
                        <1> vga_mode_tbl_ptr:
                        <1>       ; 25/07/2016
00006AE6 [466B0000]     <1>       dd     vga_mode_03h
00006AEA [466B0000]     <1>       dd     vga_mode_03h ; mode 02h -> mode 03h
00006AEE [866B0000]     <1>       dd     vga_mode_01h
00006AF2 [866B0000]     <1>       dd     vga_mode_01h ; mode 00h -> mode 01h
                        <1>       ;dd    vga_mode_07h
00006AF6 [466B0000]     <1>       dd     vga_mode_03h ; mode 07h -> mode 03h
00006AFA [C66B0000]     <1>       dd     vga_mode_04h
00006AFE [C66B0000]     <1>       dd     vga_mode_04h ; mode 05h -> mode 04h
00006B02 [066C0000]     <1>       dd     vga_mode_06h
00006B06 [466C0000]     <1>       dd     vga_mode_13h
00006B0A [866C0000]     <1>       dd     vga_mode_F0h
00006B0E [C66C0000]     <1>       dd     vga_mode_12h
00006B12 [066D0000]     <1>       dd     vga_mode_6Ah
00006B16 [466D0000]     <1>       dd     vga_mode_0Dh
00006B1A [866D0000]     <1>       dd     vga_mode_0Eh
00006B1E [C66D0000]     <1>       dd     vga_mode_10h
00006B22 [066E0000]     <1>       dd     vga_mode_11h
                        <1>
                        <1> vga_memmodel:
```

```
                                <1>       ; 25/07/2016
                                <1>       ; 07/07/2016
                                <1>       CTEXT  equ 0
                                <1>       ;MTEXT equ 1
                                <1>       MTEXT  equ 0 ; mode 07h -> mode 03h
                                <1>       CGA    equ 2
                                <1>       LINEAR8 equ 5
                                <1>       PLANAR4 equ 4
                                <1>       PLANAR1 equ 3
00006B26 0000000000020202       <1>       db     CTEXT, CTEXT, CTEXT, CTEXT, MTEXT, CGA, CGA, CGA
                                <1> vga_g_memmodel: ; 31/07/2016
00006B2E 0504040404040403       <1>       db     LINEAR8, PLANAR4, PLANAR4, PLANAR4, PLANAR4, PLANAR4, PLANAR4, PLANAR1
                                <1> ;vga_pixbits:
                                <1> ;     ; 25/07/2016
                                <1> ;     ; 08/07/2016
                                <1> ;     db     4, 4, 4, 4, 4, 2, 2, 1, 8, 4, 4, 4, 4, 4, 4, 1
                                <1> vga_dac_s:
00006B36 020202020001010103-    <1>       db     2, 2, 2, 2, 0, 1, 1, 1, 3, 3, 2, 2, 1, 1, 2, 2
00006B3F 03020201010202         <1>
                                <1>                                         ; (vgatables.h, VGAMODES, dac)
                                <1>                                         ; 17/11/2020
                                <1> vga_params:
                                <1>       ; 23/11/2020
                                <1>       ; 16/11/2020
                                <1>       ; 09/11/2020, 10/11/2020, 11/11/2020 (TRDOS 386 v2.0.3)
                                <1>       ; 25/07/2016
                                <1>       ; 19/07/2016
                                <1>       ; 03/07/2016
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios Developers Team (2001-2008)
                                <1>       ; 'vgatables.h'
                                <1>       ; Oracle VirtualBox 5.0.24 VGABios Source Code
                                <1>       ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
                                <1>
                                <1>       ; 09/11/2020
                                <1>       ; Block Structure of Video Parameter Table
                                <1>       ;
                                <1>       ; Offset     # Bytes      Contents
                                <1>       ;
                                <1>       ;  0         1            # columns
                                <1>       ;  1         1            # rows - 1
                                <1>       ;  2         1            Pixels/character
                                <1>       ;  3-4       2            Page length
                                <1>       ;  5-8       4            Sequencer Registers
                                <1>       ;  9         1            Miscellaneous Register
                                <1>       ;  10-34     25           CRTC Registers
                                <1>       ;  35-54     20           Attribute Registers
                                <1>       ;  55-63     9            Graphics Controller Registers
                                <1>       ;
                                <1>       ; Ref: Programmer's Guide to EGA, VGA, and Super VGA cards
                                <1>       ; (Richard F. Ferraro, 1994)
                                <1>
                                <1>       ;
                                <1> vga_mode_03h:  ; mode 03h, 80*25 text, CGA colors
                                <1>       ; 11/11/2020
00006B46 5018100010             <1>       db     80, 24, 16, 00h, 10h ; tw, th-1, ch, slength (5)
00006B4B 00030002               <1>       db     00h, 03h, 00h, 02h ; sequ regs (4)
00006B4F 67                     <1>       db     67h    ; misc reg (1)
00006B50 5F4F50825581BF1F       <1>       db     5Fh, 4Fh, 50h, 82h, 55h, 81h, 0BFh, 1Fh
                                <1>       ; 09/11/2020
                                <1>       ;db    5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
00006B58 004F                   <1>       db     00h, 4Fh
                                <1> vga_p_cm_pos equ $ - vga_mode_03h
00006B5A 0D0E00000000           <1>       db     0Dh, 0Eh, 00h, 00h, 00h, 00h
00006B60 9C8E8F281F96B9A3       <1>       db     9Ch, 8Eh, 8Fh, 28h, 1Fh, 96h, 0B9h, 0A3h
00006B68 FF                     <1>       db     0FFh   ; crtc_regs (25)
00006B69 0001020304051407       <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
00006B71 38393A3B3C3D3E3F       <1>       db     38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
                                <1>       ;db    0Ch, 00h, 0Fh, 08h  ; actl regs (20)
00006B79 0C000F00               <1>       db     0Ch, 00h, 0Fh, 00h  ; 19/11/2020
00006B7D 0000000000100E0FFF     <1>       db     00h, 00h, 00h, 00h, 00h, 10h, 0Eh, 0Fh, 0FFh ; grdc regs (9)
                                <1>       ; 09/11/2020
                                <1>       ;db    00h, 00h, 00h, 00h, 00h, 10h, 0Eh, 00h, 0FFh ; grdc regs (9)
                                <1> vga_mode_01h:      ; mode 01h, 40*25 text, CGA colors
00006B86 2818100008             <1>       db     40, 24, 16, 00h, 08h ; tw, th-1, ch, slength
00006B8B 08030002               <1>       db     08h, 03h, 00h, 02h  ; sequ regs
00006B8F 67                     <1>       db     67h    ; misc reg
00006B90 2D2728902BA0BF1F       <1>       db     2Dh, 27h, 28h, 90h, 2Bh, 0A0h, 0BFh, 1Fh
00006B98 004F0D0E00000000       <1>       db     00h, 4Fh, 0Dh, 0Eh, 00h, 00h, 00h, 00h
00006BA0 9C8E8F141F96B9A3       <1>       db     9Ch, 8Eh, 8Fh, 14h, 1Fh, 96h, 0B9h, 0A3h
00006BA8 FF                     <1>       db     0FFh   ; crtc_regs
00006BA9 0001020304051407       <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
00006BB1 38393A3B3C3D3E3F       <1>       db     38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
                                <1>       ;db    0Ch, 00h, 0Fh, 08h  ; actl regs (20)
00006BB9 0C000F00               <1>       db     0Ch, 00h, 0Fh, 00h  ; 19/11/2020
00006BBD 0000000000100E0FFF     <1>       db     00h, 00h, 00h, 00h, 00h, 10h, 0Eh, 0Fh, 0FFh ; grdc regs
                                <1> ;vga_mode_07h:     ; mode 07h, 80*25 text, mono color
                                <1> ;     db     80, 24, 16, 00h, 10h  ; tw, th-1, ch, slength
                                <1> ;     db     00h, 03h, 00h, 02h ; sequ regs
                                <1> ;     db     66h    ; misc reg
                                <1> ;     db     5Fh, 4Fh, 50h, 82h, 55h, 81h, 0BFh, 1Fh
                                <1> ;     db     00h, 4Fh, 0Dh, 0Eh, 00h, 00h, 00h, 00h
                                <1> ;     db     9Ch, 8Eh, 8Fh, 28h, 0Fh, 96h, 0B9h, 0A3h
                                <1> ;     db     0FFh   ; crtc regs
                                <1> ;     db     00h, 08h, 08h, 08h, 08h, 08h, 08h, 08h
                                <1> ;     db     10h, 18h, 18h, 18h, 18h, 18h, 18h, 18h
                                <1> ;     db     0Eh, 00h, 0Fh, 08h  ; actl regs
                                <1> ;     db     00h, 00h, 00h, 00h, 00h, 10h, 0Ah, 0Fh, 0FFh ; grdc regs
                                <1> vga_mode_04h:      ; 320*200 graphics, 4 colors, CGA
                                <1>       ; 11/11/2020
00006BC6 2818080040             <1>       db     40, 24, 8, 00h, 40h ; tw, th-1, ch, slength
00006BCB 09030002               <1>       db     09h, 03h, 00h, 02h ; sequ regs
00006BCF 63                     <1>       db     63h    ; misc reg
00006BD0 2D2728902B80BF1F       <1>       db     2Dh, 27h, 28h, 90h, 2Bh, 80h, 0BFh, 1Fh
```

```
3324 00006BD8 00C1000000000000    <1>       db     00h, 0C1h, 00h, 00h, 00h, 00h, 00h, 00h
3325 00006BE0 9C8E8F140096B9A2    <1>       db     9Ch, 8Eh, 8Fh, 14h, 00h, 96h, 0B9h, 0A2h
3326 00006BE8 FF                  <1>       db     0FFh   ; crtc_regs
3327 00006BE9 0013151702040607    <1>       db     00h, 13h, 15h, 17h, 02h, 04h, 06h, 07h
3328 00006BF1 1011121314151617    <1>       db     10h, 11h, 12h, 13h, 14h, 15h, 16h, 17h
3329 00006BF9 01000300            <1>       db     01h, 00h, 03h, 00h ; actl regs
3330 00006BFD 0000000000300F0FFF  <1>       db     00h, 00h, 00h, 00h, 00h, 30h, 0Fh, 0Fh, 0FFh ; grdc regs
3331                              <1> vga_mode_06h:      ; 640*200 graphics, 2 colors, CGA
3332                              <1>       ; 11/11/2020
3333 00006C06 5018080040          <1>       db     80, 24, 8, 00h, 40h   ;   tw, th-1, ch, slength
3334 00006C0B 01010006            <1>       db     01h, 01h, 00h, 06h ; sequ regs
3335 00006C0F 63                  <1>       db     63h    ; misc reg
3336 00006C10 5F4F50825480BF1F    <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
3337 00006C18 00C1000000000000    <1>       db     00h, 0C1h, 00h, 00h, 00h, 00h, 00h, 00h
3338 00006C20 9C8E8F280096B9C2    <1>       db     9Ch, 8Eh, 8Fh, 28h, 00h, 96h, 0B9h, 0C2h
3339 00006C28 FF                  <1>       db     0FFh   ; crtc regs
3340 00006C29 0017171717171717    <1>       db     00h, 17h, 17h, 17h, 17h, 17h, 17h, 17h
3341 00006C31 1717171717171717    <1>       db     17h, 17h, 17h, 17h, 17h, 17h, 17h, 17h
3342 00006C39 01000100            <1>       db     01h, 00h, 01, 00h  ; actl regs
3343 00006C3D 0000000000000D0FFF  <1>       db     00h, 00h, 00h, 00h, 00h, 00h, 0Dh, 0Fh, 0FFh ; grdc regs
3344                              <1> vga_mode_13h:  ; mode 13h, 300*200, 256 colors, linear
3345                              <1>       ; 11/11/2020
3346                              <1>       ;db    40, 24, 8, 00h, 20h ; tw, th-1, ch, slength (5)
3347                              <1>       ; 23/11/2020 - 10/11/2020
3348 00006C46 28180800FA          <1>       db     40, 24, 8, 00h, 0FAh ; tw, th-1, ch, slength (5)
3349 00006C4B 010F000E            <1>       db     01h, 0Fh, 00h, 0Eh ; sequ regs (4)
3350 00006C4F 63                  <1>       db     63h    ; misc reg (1)
3351 00006C50 5F4F50825480BF1F    <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
3352 00006C58 0041000000000000    <1>       db     00h, 041h, 00h, 00h, 00h, 00h, 00h, 00h
3353 00006C60 9C8E8F284096B9A3    <1>       db     9Ch, 8Eh, 8Fh, 28h, 40h, 96h, 0B9h, 0A3h
3354 00006C68 FF                  <1>       db     0FFh   ; crtc regs (25)
3355 00006C69 0001020304050607    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
3356 00006C71 08090A0B0C0D0E0F    <1>       db     08h, 09h, 0Ah, 0Bh, 0Ch, 0Dh, 0Eh, 0Fh
3357 00006C79 41000F00            <1>       db     41h, 00h, 0Fh, 00h  ; actl regs (20)
3358                              <1>       ; 10/11/2020
3359                              <1>       ;db    41h, 01h, 0Fh, 13h  ; actl regs (20)
3360 00006C7D 000000000040050FFF  <1>       db     00h, 00h, 00h, 00h, 00h, 40h, 05h, 0Fh, 0FFh ; grdc regs (9)
3361                              <1> vga_mode_setl equ $ - vga_mode_13h  ; = 64
3362                              <1> vga_mode_F0h:  ; mode X ; 320*240, 256 colors, planar
3363 00006C86 2818080000          <1>       db     40, 24, 8, 00h, 00h ; tw, th-1, ch, slength
3364 00006C8B 010F0006            <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
3365 00006C8F E3                  <1>       db     0E3h   ; misc reg
3366 00006C90 5F4F508254800D3E    <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0Dh, 3Eh
3367 00006C98 0041000000000000    <1>       db     00h, 41h, 00h, 00h, 00h, 00h, 00h, 00h
3368 00006CA0 EAACDF2800E706E3    <1>       db     0EAh, 0ACh, 0DFh, 28h, 00h, 0E7h, 06h, 0E3h
3369 00006CA8 FF                  <1>       db     0FFh   ; crtc regs (25)
3370 00006CA9 0001020304050607    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
3371 00006CB1 08090A0B0C0D0E0F    <1>       db     08h, 09h, 0Ah, 0Bh, 0Ch, 0Dh, 0Eh, 0Fh
3372 00006CB9 41000F00            <1>       db     41h, 00h, 0Fh, 00h  ; actl regs
3373 00006CBD 000000000040050FFF  <1>       db     00h, 00h, 00h, 00h, 00h, 40h, 05h, 0Fh, 0FFh ; grdc regs
3374                              <1> vga_mode_12h:  ; mode 12h, 640*480, 16 colors, planar
3375                              <1>       ; 11/11/2020
3376                              <1>       ;db    80, 29, 16, 0, 0 ; tw, th-1, ch, slength
3377                              <1>       ; 09/11/2020
3378 00006CC6 501D1000A0          <1>       db     80, 29, 16, 00h, 0A0h ; tw, th-1, ch, slength
3379 00006CCB 010F0006            <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
3380 00006CCF E3                  <1>       db     0E3h   ; misc reg
3381 00006CD0 5F4F508254800B3E    <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0Bh, 3Eh
3382                              <1>       ; 09/11/2020
3383                              <1>       ;db    5Fh, 4Fh, 50h, 82h, 53h, 9Fh, 0Bh, 3Eh
3384 00006CD8 0040000000000000    <1>       db     00h, 40h, 00h, 00h, 00h, 00h, 00h, 00h
3385 00006CE0 EA8CDF2800E704E3    <1>       db     0EAh, 8Ch, 0DFh, 28h, 00h, 0E7h, 04h, 0E3h
3386                              <1>       ; 09/11/2020
3387                              <1>       ;db    0E9h, 8Bh, 0DFh, 28h, 00h, 0E7h, 04h, 0E3h
3388 00006CE8 FF                  <1>       db     0FFh   ; crtc regs
3389 00006CE9 0001020304051407    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
3390 00006CF1 38393A3B3C3D3E3F    <1>       db     38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
3391 00006CF9 01000F00            <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
3392 00006CFD 000000000000050FFF  <1>       db     00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
3393                              <1> vga_mode_6Ah:  ; mode 6Ah, 800*600, 16 colors, planar
3394                              <1>       ; 11/11/2020
3395 00006D06 6424100000          <1>       db     100, 36, 16, 00h, 00h ; tw, th-1, ch, slength
3396 00006D0B 010F0006            <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
3397 00006D0F E3                  <1>       db     0E3h   ; misc reg
3398 00006D10 7F6363836B1B72F0    <1>       db     7Fh, 63h, 63h, 83h, 6Bh, 1Bh, 72h, 0F0h
3399 00006D18 0060000000000000    <1>       db     00h, 60h, 00h, 00h, 00h, 00h, 00h, 00h
3400 00006D20 598D5732005773E3    <1>       db     59h, 8Dh, 57h, 32h, 00h, 57h, 73h, 0E3h
3401 00006D28 FF                  <1>       db     0FFh   ; crtc regs
3402 00006D29 0001020304051407    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
3403 00006D31 38393A3B3C3D3E3F    <1>       db     38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
3404 00006D39 01000F00            <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
3405 00006D3D 000000000000050FFF  <1>       db     00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
3406                              <1> vga_mode_0Dh:  ; mode 0Dh, 320*200, 16 colors, planar
3407 00006D46 2818080020          <1>       db     40, 24, 8, 00h, 20h ; tw, th-1, ch, slength
3408 00006D4B 090F0006            <1>       db     09h, 0Fh, 00h, 06h ; sequ regs
3409 00006D4F 63                  <1>       db     63h    ; misc reg
3410 00006D50 2D2728902B80BF1F    <1>       db     2Dh, 27h, 28h, 90h, 2Bh, 80h, 0BFh, 1Fh
3411 00006D58 00C0000000000000    <1>       db     00h, 0C0h, 00h, 00h, 00h, 00h, 00h, 00h
3412 00006D60 9C8E8F140096B9E3    <1>       db     9Ch, 8Eh, 8Fh, 14h, 00h, 96h, 0B9h, 0E3h
3413 00006D68 FF                  <1>       db     0FFh   ; crtc regs
3414 00006D69 0001020304050607    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
3415 00006D71 1011121314151617    <1>       db     10h, 11h, 12h, 13h, 14h, 15h, 16h, 17h
3416 00006D79 01000F00            <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
3417 00006D7D 000000000000050FFF  <1>       db     00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
3418                              <1> vga_mode_0Eh:  ; mode 0Eh, 640*200, 16 colors, planar
3419 00006D86 5018080040          <1>       db     80, 24, 8, 00h, 40h ; tw, th-1, ch, slength
3420 00006D8B 010F0006            <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
3421 00006D8F 63                  <1>       db     63h    ; misc reg
3422 00006D90 5F4F50825480BF1F    <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
3423 00006D98 00C0000000000000    <1>       db     00h, 0C0h, 00h, 00h, 00h, 00h, 00h, 00h
3424 00006DA0 9C8E8F280096B9E3    <1>       db     9Ch, 8Eh, 8Fh, 28h, 00h, 96h, 0B9h, 0E3h
3425 00006DA8 FF                  <1>       db     0FFh   ; crtc regs
3426 00006DA9 0001020304050607    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
3427 00006DB1 1011121314151617    <1>       db     10h, 11h, 12h, 13h, 14h, 15h, 16h, 17h
3428 00006DB9 01000F00            <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
```

```
3429 00006DBD 000000000000050FFF  <1>       db    00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
3430                              <1> vga_mode_10h: ; mode 10h, 640*350, 16 colors, planar
3431 00006DC6 50180E0080          <1>       db    80, 24, 14, 00h, 80h ; tw, th-1, ch, slength
3432 00006DCB 010F0006            <1>       db    01h, 0Fh, 00h, 06h ; sequ regs
3433 00006DCF A3                  <1>       db    0A3h   ; misc reg
3434 00006DD0 5F4F50825480BF1F    <1>       db    5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
3435 00006DD8 0040000000000000    <1>       db    00h, 40h, 00h, 00h, 00h, 00h, 00h, 00h
3436 00006DE0 83855D280F63BAE3    <1>       db    83h, 85h, 5Dh, 28h, 0Fh, 63h, 0BAh, 0E3h
3437 00006DE8 FF                  <1>       db    0FFh   ; crtc regs
3438 00006DE9 0001020304051407    <1>       db    00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
3439 00006DF1 38393A3B3C3D3E3F    <1>       db    38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
3440 00006DF9 01000F00            <1>       db    01h, 00h, 0Fh, 00h  ; actl regs
3441 00006DFD 000000000000050FFF  <1>       db    00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
3442                              <1> vga_mode_11h: ; mode 11h, 640*480, mono color, planar
3443                              <1>       ; 11/11/2020
3444 00006E06 501D1000A0          <1>       db    80, 29, 16, 00h, 0A0h ; tw, th-1, ch, slength
3445 00006E0B 010F0006            <1>       db    01h, 0Fh, 00h, 06h ; sequ regs
3446 00006E0F E3                  <1>       db    0E3h   ; misc reg
3447 00006E10 5F4F508254800B3E    <1>       db    5Fh, 4Fh, 50h, 82h, 54h, 80h, 0Bh, 3Eh
3448 00006E18 0040000000000000    <1>       db    00h, 40h, 00h, 00h, 00h, 00h, 00h, 00h
3449 00006E20 EA8CDF2800E704C3    <1>       db    0EAh, 8Ch, 0DFh, 28h, 00h, 0E7h, 04h, 0C3h ; 11/11/2020
3450 00006E28 FF                  <1>       db    0FFh   ; crtc regs
3451 00006E29 003F003F003F003F    <1>       db    00h, 3Fh, 00h, 3Fh, 00h, 3Fh, 00h, 3Fh
3452 00006E31 003F003F003F003F    <1>       db    00h, 3Fh, 00h, 3Fh, 00h, 3Fh, 00h, 3Fh
3453 00006E39 01000F00            <1>       db    01h, 00h, 0Fh, 00h  ; actl regs
3454 00006E3D 000000000000050FFF  <1>       db    00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
3455                              <1> end_of_vga_params:
3456                              <1>
3457                              <1> ; /// End Of VIDEO DATA ///
3458                              <1>
3459                              <1> ; 23/11/2020
3460                              <1> ; VBE 2 BOCHS/QEMU emulator extensions
3461                              <1> ;     for TRDOS 386 v2 kernel (video bios)
3462                              <1>
3463                              <1> ; vbetables.h by Volker Rupper (02/01/2020)
3464                              <1>
3465                              <1> b_vbe_modes:
3466                              <1>       ;/* standard VESA modes */
3467 00006E46 0001800290010800    <1>       dw 100h,  640, 400,  8
3468 00006E4E 01018002E0010800    <1>       dw 101h,  640, 480,  8
3469 00006E56 0301200358020800    <1>       dw 103h,  800, 600,  8
3470 00006E5E 0501000400030800    <1>       dw 105h, 1024, 768,  8
3471 00006E66 0E014001C8001000    <1>       dw 10Eh,  320, 200, 16
3472 00006E6E 0F014001C8001800    <1>       dw 10Fh,  320, 200, 24
3473 00006E76 11018002E0011000    <1>       dw 111h,  640, 480, 16
3474 00006E7E 12018002E0011800    <1>       dw 112h,  640, 480, 24
3475 00006E86 1401200358021000    <1>       dw 114h,  800, 600, 16
3476 00006E8E 1501200358021800    <1>       dw 115h,  800, 600, 24
3477 00006E96 1701000400031000    <1>       dw 117h, 1024, 768, 16
3478 00006E9E 1801000400031800    <1>       dw 118h, 1024, 768, 24
3479                              <1>
3480                              <1> ;/* BOCHS/PLEX86 'own' mode numbers */
3481 00006EA6 40014001C8002000    <1>       dw 140h,  320,  200, 32
3482 00006EAE 4101800290012000    <1>       dw 141h,  640,  400, 32
3483 00006EB6 42018002E0012000    <1>       dw 142h,  640,  480, 32
3484 00006EBE 4301200358022000    <1>       dw 143h,  800,  600, 32
3485 00006EC6 4401000400032000    <1>       dw 144h, 1024,  768, 32
3486 00006ECE 46014001C8000800    <1>       dw 146h,  320,  200,  8
3487 00006ED6 8D010005D0021000    <1>       dw 18Dh, 1280,  720, 16
3488 00006EDE 8E010005D0021800    <1>       dw 18Eh, 1280,  720, 24
3489 00006EE6 8F010005D0022000    <1>       dw 18Fh, 1280,  720, 32
3490 00006EEE 9001800738041000    <1>       dw 190h, 1920, 1080, 16
3491 00006EF6 9101800738041800    <1>       dw 191h, 1920, 1080, 24
3492 00006EFE 9201800738042000    <1>       dw 192h, 1920, 1080, 32
3493                              <1>
3494                              <1> end_of_b_vbe_modes:
3495                              <1>
3496                              <1> MA1 equ VBE_MODE_ATTRIBUTE_SUPPORTED
3497                              <1> MA2 equ VBE_MODE_ATTRIBUTE_EXTENDED_INFO_AVAILABLE
3498                              <1> MA3 equ VBE_MODE_ATTRIBUTE_COLOR_MODE
3499                              <1> MA4 equ VBE_MODE_ATTRIBUTE_LINEAR_FRAME_BUFFER_MODE
3500                              <1> MA5 equ     VBE_MODE_ATTRIBUTE_GRAPHICS_MODE
3501                              <1>
3502                              <1> MODE_ATTRIBUTES equ MA1|MA2|MA3|MA4|MA5
3503                              <1>
3504                              <1> WA1 equ VBE_WINDOW_ATTRIBUTE_RELOCATABLE
3505                              <1> WA2 equ     VBE_WINDOW_ATTRIBUTE_READABLE
3506                              <1> WA3 equ VBE_WINDOW_ATTRIBUTE_WRITEABLE
3507                              <1>
3508                              <1> WINA_ATTRIBUTES equ WA1|WA2|WA3
3509                              <1>
3510                              <1> ; 24/11/2020
3511                              <1>
3512                              <1> %if 0
3513                              <1>
3514                              <1> MODE_INFO_LIST:
3515                              <1>
3516                              <1> ; 24/11/2020
3517                              <1> ; '%if 0' disables 24 mode info tables here, until %endif
3518                              <1> ; ('set_mode_info_list' will set only 1 list for selected mode)
3519                              <1> ; (Purpose: To save about 1 KB kernel size by removing fixed data)
3520                              <1>
3521                              <1>                     dw    0100h ; 640x400x8
3522                              <1> ModeAttributes1:    dw    MODE_ATTRIBUTES
3523                              <1> WinAAttributes1:    db    WINA_ATTRIBUTES
3524                              <1> WinBAttributes1:    db    0
3525                              <1> WinGranularity1:    dw    VBE_DISPI_BANK_SIZE_KB
3526                              <1> WinSize1:           dw    VBE_DISPI_BANK_SIZE_KB
3527                              <1> WinASegment1:       dw    VGAMEM_GRAPH
3528                              <1> WinBSegment1:       dw    0000h
3529                              <1> WinFuncPtr1:        dd    0
3530                              <1> BytesPerScanLine1:          dw    640
3531                              <1> XResolution1:               dw    640
3532                              <1> YResolution1:               dw    400
3533                              <1> XCharSize1:         db    8
```

```
3534                            <1> YCharSize1:         db      16
3535                            <1> NumberOfPlanes1:    db      1
3536                            <1> BitsPerPixel1:      db      8
3537                            <1> NumberOfBanks1:     db      4
3538                            <1> MemoryModel1:       db      VBE_MEMORYMODEL_PACKED_PIXEL
3539                            <1> BankSize1:          db      0
3540                            <1> NumberOfImagePages1:        db      64
3541                            <1> Reserved_page1:             db      0
3542                            <1> RedMaskSize1:               db      0
3543                            <1> RedFieldPosition1:          db      0
3544                            <1> GreenMaskSize1:             db      0
3545                            <1> GreenFieldPosition1:        db      0
3546                            <1> BlueMaskSize1:              db      0
3547                            <1> BlueFieldPosition1:         db      0
3548                            <1> RsvdMaskSize1:              db      0
3549                            <1> RsvdFieldPosition1:         db      0
3550                            <1> DirectColorModeInfo1:       db      0
3551                            <1> PhysBasePtr1:               dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
3552                            <1> OffScreenMemOffset1:        dd      0
3553                            <1> OffScreenMemSize1:          dw      0
3554                            <1> LinBytesPerScanLine1:       dw      640
3555                            <1> BnkNumberOfPages1:          db      0
3556                            <1> LinNumberOfPages1:          db      0
3557                            <1> LinRedMaskSize1:            db      0
3558                            <1> LinRedFieldPosition1:       db      0
3559                            <1> LinGreenMaskSize1:          db      0
3560                            <1> LinGreenFieldPosition1:     db      0
3561                            <1> LinBlueMaskSize1:           db      0
3562                            <1> LinBlueFieldPosition1:      db      0
3563                            <1> LinRsvdMaskSize1:           db      0
3564                            <1> LinRsvdFieldPosition1:      db      0
3565                            <1> MaxPixelClock1:             dd      0
3566                            <1>
3567                            <1>                     dw      0101h ; 640x480x8
3568                            <1> ModeAttributes2:    dw      MODE_ATTRIBUTES
3569                            <1> WinAAttributes2:    db      WINA_ATTRIBUTES
3570                            <1> WinBAttributes2:    db      0
3571                            <1> WinGranularity2:    dw      VBE_DISPI_BANK_SIZE_KB
3572                            <1> WinSize2:           dw      VBE_DISPI_BANK_SIZE_KB
3573                            <1> WinASegment2:       dw      VGAMEM_GRAPH
3574                            <1> WinBSegment2:       dw      0000h
3575                            <1> WinFuncPtr2:        dd      0
3576                            <1> BytesPerScanLine2:  dw      640
3577                            <1> XResolution2:       dw      640
3578                            <1> YResolution2:       dw      480
3579                            <1> XCharSize2:         db      8
3580                            <1> YCharSize2:         db      16
3581                            <1> NumberOfPlanes2:    db      1
3582                            <1> BitsPerPixel2:      db      8
3583                            <1> NumberOfBanks2:     db      5
3584                            <1> MemoryModel2:       db      VBE_MEMORYMODEL_PACKED_PIXEL
3585                            <1> BankSize2:          db      0
3586                            <1> NumberOfImagePages2:        db      53
3587                            <1> Reserved_page2:             db      0
3588                            <1> RedMaskSize2:               db      0
3589                            <1> RedFieldPosition2:          db      0
3590                            <1> GreenMaskSize2:             db      0
3591                            <1> GreenFieldPosition2:        db      0
3592                            <1> BlueMaskSize2:              db      0
3593                            <1> BlueFieldPosition2:         db      0
3594                            <1> RsvdMaskSize2:              db      0
3595                            <1> RsvdFieldPosition2:         db      0
3596                            <1> DirectColorModeInfo2:       db      0
3597                            <1> PhysBasePtr2:               dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
3598                            <1> OffScreenMemOffset2:        dd      0
3599                            <1> OffScreenMemSize2:          dw      0
3600                            <1> LinBytesPerScanLine2:       dw      640
3601                            <1> BnkNumberOfPages2:          db      0
3602                            <1> LinNumberOfPages2:          db      0
3603                            <1> LinRedMaskSize2:            db      0
3604                            <1> LinRedFieldPosition2:       db      0
3605                            <1> LinGreenMaskSize2:          db      0
3606                            <1> LinGreenFieldPosition2:     db      0
3607                            <1> LinBlueMaskSize2:           db      0
3608                            <1> LinBlueFieldPosition2:      db      0
3609                            <1> LinRsvdMaskSize2:           db      0
3610                            <1> LinRsvdFieldPosition2:      db      0
3611                            <1> MaxPixelClock2:             dd      0
3612                            <1>
3613                            <1>                     dw      0103h ; 800x600x8
3614                            <1> ModeAttributes3:    dw      MODE_ATTRIBUTES
3615                            <1> WinAAttributes3:    db      WINA_ATTRIBUTES
3616                            <1> WinBAttributes3:    db      0
3617                            <1> WinGranularity3:    dw      VBE_DISPI_BANK_SIZE_KB
3618                            <1> WinSize3:           dw      VBE_DISPI_BANK_SIZE_KB
3619                            <1> WinASegment3:       dw      VGAMEM_GRAPH
3620                            <1> WinBSegment3:       dw      0000h
3621                            <1> WinFuncPtr3:        dd      0
3622                            <1> BytesPerScanLine3:  dw      800
3623                            <1> XResolution3:       dw      800
3624                            <1> YResolution3:       dw      600
3625                            <1> XCharSize3:         db      8
3626                            <1> YCharSize3:         db      16
3627                            <1> NumberOfPlanes3:    db      1
3628                            <1> BitsPerPixel3:      db      8
3629                            <1> NumberOfBanks3:     db      8
3630                            <1> MemoryModel3:       db      VBE_MEMORYMODEL_PACKED_PIXEL
3631                            <1> BankSize3:          db      0
3632                            <1> NumberOfImagePages3:        db      33
3633                            <1> Reserved_page3:             db      0
3634                            <1> RedMaskSize3:               db      0
3635                            <1> RedFieldPosition3:          db      0
3636                            <1> GreenMaskSize3:             db      0
3637                            <1> GreenFieldPosition3:        db      0
3638                            <1> BlueMaskSize3:              db      0
```

```
3639                            <1> BlueFieldPosition3:      db     0
3640                            <1> RsvdMaskSize3:           db     0
3641                            <1> RsvdFieldPosition3:      db     0
3642                            <1> DirectColorModeInfo3:    db     0
3643                            <1> PhysBasePtr3:            dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
3644                            <1> OffScreenMemOffset3:     dd     0
3645                            <1> OffScreenMemSize3:       dw     0
3646                            <1> LinBytesPerScanLine3:    dw     800
3647                            <1> BnkNumberOfPages3:       db     0
3648                            <1> LinNumberOfPages3:       db     0
3649                            <1> LinRedMaskSize3:         db     0
3650                            <1> LinRedFieldPosition3:    db     0
3651                            <1> LinGreenMaskSize3:       db     0
3652                            <1> LinGreenFieldPosition:   db     0
3653                            <1> LinBlueMaskSize:         db     0
3654                            <1> LinBlueFieldPosition:    db     0
3655                            <1> LinRsvdMaskSize:         db     0
3656                            <1> LinRsvdFieldPosition:    db     0
3657                            <1> MaxPixelClock:           dd     0
3658                            <1>
3659                            <1>                  dw     0105h ; 1024x768x8
3660                            <1> ModeAttributes4:  dw     MODE_ATTRIBUTES
3661                            <1> WinAAttributes4:  db     WINA_ATTRIBUTES
3662                            <1> WinBAttributes4:  db     0
3663                            <1> WinGranularity4:  dw     VBE_DISPI_BANK_SIZE_KB
3664                            <1> WinSize4:         dw     VBE_DISPI_BANK_SIZE_KB
3665                            <1> WinASegment4:     dw     VGAMEM_GRAPH
3666                            <1> WinBSegment4:     dw     0000h
3667                            <1> WinFuncPtr4:      dd     0
3668                            <1> BytesPerScanLine4: dw    1024
3669                            <1> XResolution4:     dw     1024
3670                            <1> YResolution4:     dw     768
3671                            <1> XCharSize4:       db     8
3672                            <1> YCharSize4:       db     16
3673                            <1> NumberOfPlanes4:  db     1
3674                            <1> BitsPerPixel4:    db     8
3675                            <1> NumberOfBanks4:   db     12
3676                            <1> MemoryModel4:     db     VBE_MEMORYMODEL_PACKED_PIXEL
3677                            <1> BankSize4:        db     0
3678                            <1> NumberOfImagePages4:     db     20
3679                            <1> Reserved_page4:          db     0
3680                            <1> RedMaskSize4:            db     0
3681                            <1> RedFieldPosition4:       db     0
3682                            <1> GreenMaskSize4:          db     0
3683                            <1> GreenFieldPosition4:     db     0
3684                            <1> BlueMaskSize4:           db     0
3685                            <1> BlueFieldPosition4:      db     0
3686                            <1> RsvdMaskSize4:           db     0
3687                            <1> RsvdFieldPosition4:      db     0
3688                            <1> DirectColorModeInfo4:    db     0
3689                            <1> PhysBasePtr4:            dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
3690                            <1> OffScreenMemOffset4:     dd     0
3691                            <1> OffScreenMemSize4:       dw     0
3692                            <1> LinBytesPerScanLine4:    dw     1024
3693                            <1> BnkNumberOfPages4:       db     0
3694                            <1> LinNumberOfPages4:       db     0
3695                            <1> LinRedMaskSize4:         db     0
3696                            <1> LinRedFieldPosition4:    db     0
3697                            <1> LinGreenMaskSize4:       db     0
3698                            <1> LinGreenFieldPosition4:  db     0
3699                            <1> LinBlueMaskSize4:        db     0
3700                            <1> LinBlueFieldPosition4:   db     0
3701                            <1> LinRsvdMaskSize4:        db     0
3702                            <1> LinRsvdFieldPosition4:   db     0
3703                            <1> MaxPixelClock4:          dd     0
3704                            <1>
3705                            <1>                  dw     010Eh ; 320x200x16
3706                            <1> ModeAttributes5:  dw     MODE_ATTRIBUTES
3707                            <1> WinAAttributes5:  db     WINA_ATTRIBUTES
3708                            <1> WinBAttributes5:  db     0
3709                            <1> WinGranularity5:  dw     VBE_DISPI_BANK_SIZE_KB
3710                            <1> WinSize5:         dw     VBE_DISPI_BANK_SIZE_KB
3711                            <1> WinASegment5:     dw     VGAMEM_GRAPH
3712                            <1> WinBSegment5:     dw     0000h
3713                            <1> WinFuncPtr5:      dd     0
3714                            <1> BytesPerScanLine5: dw    640
3715                            <1> XResolution5:     dw     320
3716                            <1> YResolution5:     dw     200
3717                            <1> XCharSize5:       db     8
3718                            <1> YCharSize5:       db     16
3719                            <1> NumberOfPlanes5:  db     1
3720                            <1> BitsPerPixel5:    db     16
3721                            <1> NumberOfBanks5:   db     2
3722                            <1> MemoryModel5:     db     VBE_MEMORYMODEL_DIRECT_COLOR
3723                            <1> BankSize5:        db     0
3724                            <1> NumberOfImagePages5:     db     130
3725                            <1> Reserved_page5:          db     0
3726                            <1> RedMaskSize5:            db     5
3727                            <1> RedFieldPosition5:       db     11
3728                            <1> GreenMaskSize5:          db     6
3729                            <1> GreenFieldPosition5:     db     5
3730                            <1> BlueMaskSize5:           db     5
3731                            <1> BlueFieldPosition5:      db     0
3732                            <1> RsvdMaskSize5:           db     0
3733                            <1> RsvdFieldPosition5:      db     0
3734                            <1> DirectColorModeInfo5:    db     0
3735                            <1> PhysBasePtr5:            dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
3736                            <1> OffScreenMemOffset5:     dd     0
3737                            <1> OffScreenMemSize5:       dw     0
3738                            <1> LinBytesPerScanLine5:    dw     640
3739                            <1> BnkNumberOfPages5:       db     0
3740                            <1> LinNumberOfPages5:       db     0
3741                            <1> LinRedMaskSize5:         db     5
3742                            <1> LinRedFieldPosition5:    db     11
3743                            <1> LinGreenMaskSize5:       db     6
```

```
3744                            <1> LinGreenFieldPosition5:   db     5
3745                            <1> LinBlueMaskSize5:         db     5
3746                            <1> LinBlueFieldPosition5:    db     0
3747                            <1> LinRsvdMaskSize5:         db     0
3748                            <1> LinRsvdFieldPosition5:    db     0
3749                            <1> MaxPixelClock5:           dd     0
3750                            <1>
3751                            <1>                     dw     010Fh ; 320x200x24
3752                            <1> ModeAttributes6:    dw     MODE_ATTRIBUTES
3753                            <1> WinAAttributes6:    db     WINA_ATTRIBUTES
3754                            <1> WinBAttributes6:    db     0
3755                            <1> WinGranularity6:    dw     VBE_DISPI_BANK_SIZE_KB
3756                            <1> WinSize6:           dw     VBE_DISPI_BANK_SIZE_KB
3757                            <1> WinASegment6:       dw     VGAMEM_GRAPH
3758                            <1> WinBSegment6:       dw     0000h
3759                            <1> WinFuncPtr6:        dd     0
3760                            <1> BytesPerScanLine6:  dw     960
3761                            <1> XResolution6:       dw     320
3762                            <1> YResolution6:       dw     200
3763                            <1> XCharSize6:         db     8
3764                            <1> YCharSize6:         db     16
3765                            <1> NumberOfPlanes6:    db     1
3766                            <1> BitsPerPixel6:      db     24
3767                            <1> NumberOfBanks6:     db     3
3768                            <1> MemoryModel6:       db     VBE_MEMORYMODEL_DIRECT_COLOR
3769                            <1> BankSize6:          db     0
3770                            <1> NumberOfImagePages6:      db     86
3771                            <1> Reserved_page6:           db     0
3772                            <1> RedMaskSize6:             db     8
3773                            <1> RedFieldPosition6:        db     16
3774                            <1> GreenMaskSize6:           db     8
3775                            <1> GreenFieldPosition6:      db     8
3776                            <1> BlueMaskSize6:            db     8
3777                            <1> BlueFieldPosition6:       db     0
3778                            <1> RsvdMaskSize6:            db     0
3779                            <1> RsvdFieldPosition6:       db     0
3780                            <1> DirectColorModeInfo6:     db     0
3781                            <1> PhysBasePtr6:             dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
3782                            <1> OffScreenMemOffset6:      dd     0
3783                            <1> OffScreenMemSize6:        dw     0
3784                            <1> LinBytesPerScanLine6:     dw     960
3785                            <1> BnkNumberOfPages6:        db     0
3786                            <1> LinNumberOfPages6:        db     0
3787                            <1> LinRedMaskSize6:          db     8
3788                            <1> LinRedFieldPosition6:     db     16
3789                            <1> LinGreenMaskSize6:        db     8
3790                            <1> LinGreenFieldPosition6:   db     8
3791                            <1> LinBlueMaskSize6:         db     8
3792                            <1> LinBlueFieldPosition6:    db     0
3793                            <1> LinRsvdMaskSize6:         db     0
3794                            <1> LinRsvdFieldPosition6:    db     0
3795                            <1> MaxPixelClock6:           dd     0
3796                            <1>
3797                            <1>                     dw     0111h ; 640x480x16
3798                            <1> ModeAttributes7:    dw     MODE_ATTRIBUTES
3799                            <1> WinAAttributes7:    db     WINA_ATTRIBUTES
3800                            <1> WinBAttributes7:    db     0
3801                            <1> WinGranularity7:    dw     VBE_DISPI_BANK_SIZE_KB
3802                            <1> WinSize7:           dw     VBE_DISPI_BANK_SIZE_KB
3803                            <1> WinASegment7:       dw     VGAMEM_GRAPH
3804                            <1> WinBSegment7:       dw     0000h
3805                            <1> WinFuncPtr7:        dd     0
3806                            <1> BytesPerScanLine7:  dw     1280
3807                            <1> XResolution7:       dw     640
3808                            <1> YResolution7:       dw     480
3809                            <1> XCharSize7:         db     8
3810                            <1> YCharSize7:         db     16
3811                            <1> NumberOfPlanes7:    db     1
3812                            <1> BitsPerPixel7:      db     16
3813                            <1> NumberOfBanks7:     db     10
3814                            <1> MemoryModel7:       db     VBE_MEMORYMODEL_DIRECT_COLOR
3815                            <1> BankSize7:          db     0
3816                            <1> NumberOfImagePages7:      db     26
3817                            <1> Reserved_page7:           db     0
3818                            <1> RedMaskSize7:             db     5
3819                            <1> RedFieldPosition7:        db     11
3820                            <1> GreenMaskSize7:           db     6
3821                            <1> GreenFieldPosition7:      db     5
3822                            <1> BlueMaskSize7:            db     5
3823                            <1> BlueFieldPosition7:       db     0
3824                            <1> RsvdMaskSize7:            db     0
3825                            <1> RsvdFieldPosition7:       db     0
3826                            <1> DirectColorModeInfo7:     db     0
3827                            <1> PhysBasePtr7:             dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS,
3828                            <1> OffScreenMemOffset7:      dd     0
3829                            <1> OffScreenMemSize7:        dw     0
3830                            <1> LinBytesPerScanLine7:     dw     1280
3831                            <1> BnkNumberOfPages7:        db     0
3832                            <1> LinNumberOfPages7:        db     0
3833                            <1> LinRedMaskSize7:          db     5
3834                            <1> LinRedFieldPosition7:     db     11
3835                            <1> LinGreenMaskSize7:        db     6
3836                            <1> LinGreenFieldPosition7:   db     5
3837                            <1> LinBlueMaskSize7:         db     5
3838                            <1> LinBlueFieldPosition7:    db     0
3839                            <1> LinRsvdMaskSize7:         db     0
3840                            <1> LinRsvdFieldPosition7:    db     0
3841                            <1> MaxPixelClock7:           dd     0
3842                            <1>
3843                            <1>                     dw     0112h ; 640x480x24
3844                            <1> ModeAttributes8:    dw     MODE_ATTRIBUTES
3845                            <1> WinAAttributes8:    db     WINA_ATTRIBUTES
3846                            <1> WinBAttributes8:    db     0
3847                            <1> WinGranularity8:    dw     VBE_DISPI_BANK_SIZE_KB
3848                            <1> WinSize8:           dw     VBE_DISPI_BANK_SIZE_KB
```

```
3849                            <1> WinASegment8:       dw      VGAMEM_GRAPH
3850                            <1> WinBSegment8:       dw      0000h
3851                            <1> WinFuncPtr8:        dd      0
3852                            <1> BytesPerScanLine8:  dw      1920
3853                            <1> XResolution8:       dw      640
3854                            <1> YResolution8:       dw      480
3855                            <1> XCharSize8:         db      8
3856                            <1> YCharSize8:         db      16
3857                            <1> NumberOfPlanes8:    db      1
3858                            <1> BitsPerPixel8:      db      24
3859                            <1> NumberOfBanks8:     db      15
3860                            <1> MemoryModel8:       db      VBE_MEMORYMODEL_DIRECT_COLOR
3861                            <1> BankSize8:          db      0
3862                            <1> NumberOfImagePages8:        db      17
3863                            <1> Reserved_page8:             db      0
3864                            <1> RedMaskSize8:               db      8
3865                            <1> RedFieldPosition8:          db      16
3866                            <1> GreenMaskSize8:             db      8
3867                            <1> GreenFieldPosition8:        db      8
3868                            <1> BlueMaskSize8:              db      8
3869                            <1> BlueFieldPosition8:         db      0
3870                            <1> RsvdMaskSize8:              db      0
3871                            <1> RsvdFieldPosition8:         db      0
3872                            <1> DirectColorModeInfo8:       db      0
3873                            <1> PhysBasePtr8:               dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
3874                            <1> OffScreenMemOffset8:        dd      0
3875                            <1> OffScreenMemSize8:          dw      0
3876                            <1> LinBytesPerScanLine8:       dw      1920
3877                            <1> BnkNumberOfPages8:          db      0
3878                            <1> LinNumberOfPages8:          db      0
3879                            <1> LinRedMaskSize8:            db      8
3880                            <1> LinRedFieldPosition8:       db      16
3881                            <1> LinGreenMaskSize8:          db      8
3882                            <1> LinGreenFieldPosition8:     db      8
3883                            <1> LinBlueMaskSize8:           db      8
3884                            <1> LinBlueFieldPosition8:      db      0
3885                            <1> LinRsvdMaskSize8:           db      0
3886                            <1> LinRsvdFieldPosition8:      db      0
3887                            <1> MaxPixelClock8:             dd      0
3888                            <1>
3889                            <1>                     dw      0114h ; 800x600x16
3890                            <1> ModeAttributes9:    dw      MODE_ATTRIBUTES
3891                            <1> WinAAttributes9:    db      WINA_ATTRIBUTES
3892                            <1> WinBAttributes9:    db      0
3893                            <1> WinGranularity9:    dw      VBE_DISPI_BANK_SIZE_KB
3894                            <1> WinSize9:           dw      VBE_DISPI_BANK_SIZE_KB
3895                            <1> WinASegment9:       dw      VGAMEM_GRAPH
3896                            <1> WinBSegment9:       dw      0000h
3897                            <1> WinFuncPtr9:        dd      0
3898                            <1> BytesPerScanLine9:  dw      1600
3899                            <1> XResolution9:       dw      800
3900                            <1> YResolution9:       dw      600
3901                            <1> XCharSize9:         db      8
3902                            <1> YCharSize9:         db      16
3903                            <1> NumberOfPlanes9:    db      1
3904                            <1> BitsPerPixel9:      db      16
3905                            <1> NumberOfBanks9:     db      15
3906                            <1> MemoryModel9:       db      VBE_MEMORYMODEL_DIRECT_COLOR
3907                            <1> BankSize9:          db      0
3908                            <1> NumberOfImagePages9:        db      16
3909                            <1> Reserved_page9:             db      0
3910                            <1> RedMaskSize9:               db      5
3911                            <1> RedFieldPosition9:          db      11
3912                            <1> GreenMaskSize9:             db      6
3913                            <1> GreenFieldPosition9:        db      5
3914                            <1> BlueMaskSize9:              db      5
3915                            <1> BlueFieldPosition9:         db      0
3916                            <1> RsvdMaskSize9:              db      0
3917                            <1> RsvdFieldPosition9:         db      0
3918                            <1> DirectColorModeInfo9:       db      0
3919                            <1> PhysBasePtr9:               dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS
3920                            <1> OffScreenMemOffset9:        dd      0
3921                            <1> OffScreenMemSize9:          dw      0
3922                            <1> LinBytesPerScanLine9:       dw      1600
3923                            <1> BnkNumberOfPages9:          db      0
3924                            <1> LinNumberOfPages9:          db      0
3925                            <1> LinRedMaskSize9:            db      5
3926                            <1> LinRedFieldPosition9:       db      11
3927                            <1> LinGreenMaskSize9:          db      6
3928                            <1> LinGreenFieldPosition9:     db      5
3929                            <1> LinBlueMaskSize9:           db      5
3930                            <1> LinBlueFieldPosition9:      db      0
3931                            <1> LinRsvdMaskSize9:           db      0
3932                            <1> LinRsvdFieldPosition9:      db      0
3933                            <1> MaxPixelClock9:             dd      0
3934                            <1>
3935                            <1>                     dw      0115h ; 800x600x24
3936                            <1> ModeAttributes10:   dw      MODE_ATTRIBUTES
3937                            <1> WinAAttributes10:   db      WINA_ATTRIBUTES
3938                            <1> WinBAttributes10:   db      0
3939                            <1> WinGranularity10:   dw      VBE_DISPI_BANK_SIZE_KB
3940                            <1> WinSize10:          dw      VBE_DISPI_BANK_SIZE_KB
3941                            <1> WinASegment10:      dw      VGAMEM_GRAPH
3942                            <1> WinBSegment10:      dw      0000h
3943                            <1> WinFuncPtr10:       dd      0
3944                            <1> BytesPerScanLine10: dw      2400
3945                            <1> XResolution10:      dw      800
3946                            <1> YResolution10:      dw      600
3947                            <1> XCharSize10:        db      8
3948                            <1> YCharSize10:        db      16
3949                            <1> NumberOfPlanes10:   db      1
3950                            <1> BitsPerPixel10:     db      24
3951                            <1> NumberOfBanks10:    db      22
3952                            <1> MemoryModel10:      db      VBE_MEMORYMODEL_DIRECT_COLOR
3953                            <1> BankSize10:         db      0
```

```
3954                            <1> NumberOfImagePages10:     db    10
3955                            <1> Reserved_page10:          db    0
3956                            <1> RedMaskSize10:            db    8
3957                            <1> RedFieldPosition10:       db    16
3958                            <1> GreenMaskSize10:          db    8
3959                            <1> GreenFieldPosition10:     db    8
3960                            <1> BlueMaskSize10:           db    8
3961                            <1> BlueFieldPosition10:      db    0
3962                            <1> RsvdMaskSize10:           db    0
3963                            <1> RsvdFieldPosition10:      db    0
3964                            <1> DirectColorModeInfo10:    db    0
3965                            <1> PhysBasePtr10:            dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
3966                            <1> OffScreenMemOffset10:     dd    0
3967                            <1> OffScreenMemSize10:       dw    0
3968                            <1> LinBytesPerScanLine10:    dw    2400
3969                            <1> BnkNumberOfPages10:       db    0
3970                            <1> LinNumberOfPages10:       db    0
3971                            <1> LinRedMaskSize10:         db    8
3972                            <1> LinRedFieldPosition10:    db    16
3973                            <1> LinGreenMaskSize10:       db    8
3974                            <1> LinGreenFieldPosition10:  db    8
3975                            <1> LinBlueMaskSize10:        db    8
3976                            <1> LinBlueFieldPosition10:   db    0
3977                            <1> LinRsvdMaskSize10:        db    0
3978                            <1> LinRsvdFieldPosition10:   db    0
3979                            <1> MaxPixelClock10:          dd    0
3980                            <1>
3981                            <1>                   dw    0117h ; 1024x768x16
3982                            <1> ModeAttributes11: dw    MODE_ATTRIBUTES
3983                            <1> WinAAttributes11: db    WINA_ATTRIBUTES
3984                            <1> WinBAttributes11: db    0
3985                            <1> WinGranularity11: dw    VBE_DISPI_BANK_SIZE_KB
3986                            <1> WinSize11:        dw    VBE_DISPI_BANK_SIZE_KB
3987                            <1> WinASegment11:    dw    VGAMEM_GRAPH
3988                            <1> WinBSegment11:    dw    0000h
3989                            <1> WinFuncPtr11:     dd    0
3990                            <1> BytesPerScanLine11: dw  2048
3991                            <1> XResolution11:    dw    1024
3992                            <1> YResolution11:    dw    768
3993                            <1> XCharSize11:      db    8
3994                            <1> YCharSize11:      db    16
3995                            <1> NumberOfPlanes11: db    1
3996                            <1> BitsPerPixel11:   db    16
3997                            <1> NumberOfBanks11:  db    24
3998                            <1> MemoryModel11:    db    VBE_MEMORYMODEL_DIRECT_COLOR
3999                            <1> BankSize11:       db    0
4000                            <1> NumberOfImagePages11:     db    9
4001                            <1> Reserved_page11:          db    0
4002                            <1> RedMaskSize11:            db    5
4003                            <1> RedFieldPosition11:       db    11
4004                            <1> GreenMaskSize11:          db    6
4005                            <1> GreenFieldPosition11:     db    5
4006                            <1> BlueMaskSize11:           db    5
4007                            <1> BlueFieldPosition11:      db    0
4008                            <1> RsvdMaskSize11:           db    0
4009                            <1> RsvdFieldPosition11:      db    0
4010                            <1> DirectColorModeInfo11:    db    0
4011                            <1> PhysBasePtr11:            dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS,
4012                            <1> OffScreenMemOffset11:     dd    0
4013                            <1> OffScreenMemSize11:       dw    0
4014                            <1> LinBytesPerScanLine11:    dw    2048
4015                            <1> BnkNumberOfPages11:       db    0
4016                            <1> LinNumberOfPages11:       db    0
4017                            <1> LinRedMaskSize11:         db    5
4018                            <1> LinRedFieldPosition11:    db    11
4019                            <1> LinGreenMaskSize11:       db    6
4020                            <1> LinGreenFieldPosition11:db      5
4021                            <1> LinBlueMaskSize11:        db    5
4022                            <1> LinBlueFieldPosition11:   db    0
4023                            <1> LinRsvdMaskSize11:        db    0
4024                            <1> LinRsvdFieldPosition11:   db    0
4025                            <1> MaxPixelClock11:          dd    0
4026                            <1>
4027                            <1>                   dw    0118h ; 1024x768x24
4028                            <1> ModeAttributes12: dw    MODE_ATTRIBUTES
4029                            <1> WinAAttributes12: db    WINA_ATTRIBUTES
4030                            <1> WinBAttributes12: db    0
4031                            <1> WinGranularity12: dw    VBE_DISPI_BANK_SIZE_KB
4032                            <1> WinSize12:        dw    VBE_DISPI_BANK_SIZE_KB
4033                            <1> WinASegment12:    dw    VGAMEM_GRAPH
4034                            <1> WinBSegment12:    dw    0000h
4035                            <1> WinFuncPtr12:     dd    0
4036                            <1> BytesPerScanLine12: dw  3072
4037                            <1> XResolution12:    dw    1024
4038                            <1> YResolution12:    dw    768
4039                            <1> XCharSize12:      db    8
4040                            <1> YCharSize12:      db    16
4041                            <1> NumberOfPlanes12: db    1
4042                            <1> BitsPerPixel12:   db    24
4043                            <1> NumberOfBanks12:  db    36
4044                            <1> MemoryModel12:    db    VBE_MEMORYMODEL_DIRECT_COLOR
4045                            <1> BankSize12:       db    0
4046                            <1> NumberOfImagePages12:     db    6
4047                            <1> Reserved_page12:          db    0
4048                            <1> RedMaskSize12:            db    8
4049                            <1> RedFieldPosition12:       db    16
4050                            <1> GreenMaskSize12:          db    8
4051                            <1> GreenFieldPosition12:     db    8
4052                            <1> BlueMaskSize12:           db    8
4053                            <1> BlueFieldPosition12:      db    0
4054                            <1> RsvdMaskSize12:           db    0
4055                            <1> RsvdFieldPosition12:      db    0
4056                            <1> DirectColorModeInfo12:    db    0
4057                            <1> PhysBasePtr12:            dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
4058                            <1> OffScreenMemOffset12:     dd    0
```

```
                        <1> OffScreenMemSize12:       dw     0
                        <1> LinBytesPerScanLine12:    dw     3072
                        <1> BnkNumberOfPages12:       db     0
                        <1> LinNumberOfPages12:       db     0
                        <1> LinRedMaskSize12:         db     8
                        <1> LinRedFieldPosition12:    db     16
                        <1> LinGreenMaskSize12:       db     8
                        <1> LinGreenFieldPosition12:db       8
                        <1> LinBlueMaskSize12:        db     8
                        <1> LinBlueFieldPosition12:   db     0
                        <1> LinRsvdMaskSize12:        db     0
                        <1> LinRsvdFieldPosition12:   db     0
                        <1> MaxPixelClock12:          dd     0
                        <1>
                        <1>                   dw     0140h ; 320x200x32
                        <1> ModeAttributes13:  dw     MODE_ATTRIBUTES
                        <1> WinAAttributes13:  db     WINA_ATTRIBUTES
                        <1> WinBAttributes13:  db     0
                        <1> WinGranularity13:  dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize13:         dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment13:     dw     VGAMEM_GRAPH
                        <1> WinBSegment13:     dw     0000h
                        <1> WinFuncPtr13:      dd     0
                        <1> BytesPerScanLine13: dw    1280
                        <1> XResolution13:     dw     320
                        <1> YResolution13:     dw     200
                        <1> XCharSize13:       db     8
                        <1> YCharSize13:       db     16
                        <1> NumberOfPlanes13:  db     1
                        <1> BitsPerPixel13:    db     32
                        <1> NumberOfBanks13:   db     4
                        <1> MemoryModel13:     db     VBE_MEMORYMODEL_DIRECT_COLOR
                        <1> BankSize13:        db     0
                        <1> NumberOfImagePages13:     db     64
                        <1> Reserved_page13:          db     0
                        <1> RedMaskSize13:            db     8
                        <1> RedFieldPosition13:       db     16
                        <1> GreenMaskSize13:          db     8
                        <1> GreenFieldPosition13:     db     8
                        <1> BlueMaskSize13:           db     8
                        <1> BlueFieldPosition13:      db     0
                        <1> RsvdMaskSize13:           db     8
                        <1> RsvdFieldPosition13:      db     24
                        <1> DirectColorModeInfo13:    db     VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
                        <1> PhysBasePtr13:            dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
                        <1> OffScreenMemOffset13:     dd     0
                        <1> OffScreenMemSize13:       dw     0
                        <1> LinBytesPerScanLine13:    dw     1280
                        <1> BnkNumberOfPages13:       db     0
                        <1> LinNumberOfPages13:       db     0
                        <1> LinRedMaskSize13:         db     8
                        <1> LinRedFieldPosition13:    db     16
                        <1> LinGreenMaskSize13:       db     8
                        <1> LinGreenFieldPosition13:db       8
                        <1> LinBlueMaskSize13:        db     8
                        <1> LinBlueFieldPosition13:   db     0
                        <1> LinRsvdMaskSize13:        db     8
                        <1> LinRsvdFieldPosition13:   db     24
                        <1> MaxPixelClock13:          dd     0
                        <1>
                        <1>                   dw     0141h ; 640x400x32
                        <1> ModeAttributes14:  dw     MODE_ATTRIBUTES
                        <1> WinAAttributes14:  db     WINA_ATTRIBUTES
                        <1> WinBAttributes14:  db     0
                        <1> WinGranularity14:  dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize14:         dw     VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment14:     dw     VGAMEM_GRAPH
                        <1> WinBSegment14:     dw     0000h
                        <1> WinFuncPtr14:      dd     0
                        <1> BytesPerScanLine14: dw    2560
                        <1> XResolution14:     dw     640
                        <1> YResolution14:     dw     400
                        <1> XCharSize14:       db     8
                        <1> YCharSize14:       db     16
                        <1> NumberOfPlanes14:  db     1
                        <1> BitsPerPixel14:    db     32
                        <1> NumberOfBanks14:   db     16
                        <1> MemoryModel14:     db     VBE_MEMORYMODEL_DIRECT_COLOR
                        <1> BankSize14:        db     0
                        <1> NumberOfImagePages14:     db     15
                        <1> Reserved_page14:          db     0
                        <1> RedMaskSize14:            db     8
                        <1> RedFieldPosition14:       db     16
                        <1> GreenMaskSize14:          db     8
                        <1> GreenFieldPosition14:     db     8
                        <1> BlueMaskSize14:           db     8
                        <1> BlueFieldPosition14:      db     0
                        <1> RsvdMaskSize14:           db     8
                        <1> RsvdFieldPosition14:      db     24
                        <1> DirectColorModeInfo14:    db     VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
                        <1> PhysBasePtr14:            dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
                        <1> OffScreenMemOffset14:     dd     0
                        <1> OffScreenMemSize14:       dw     0
                        <1> LinBytesPerScanLine14:    dw     2560
                        <1> BnkNumberOfPages14:       db     0
                        <1> LinNumberOfPages14:       db     0
                        <1> LinRedMaskSize14:         db     8
                        <1> LinRedFieldPosition14:    db     16
                        <1> LinGreenMaskSize14:       db     8
                        <1> LinGreenFieldPosition14:db       8
                        <1> LinBlueMaskSize14:        db     8
                        <1> LinBlueFieldPosition14:   db     0
                        <1> LinRsvdMaskSize14:        db     8
                        <1> LinRsvdFieldPosition14:   db     24
                        <1> MaxPixelClock14:          dd     0
```

```
                        <1>
                        <1>                     dw      0142 ; 640x480x32
                        <1> ModeAttributes15:   dw      MODE_ATTRIBUTES
                        <1> WinAAttributes15:   db      WINA_ATTRIBUTES
                        <1> WinBAttributes15:   db      0
                        <1> WinGranularity15:   dw      VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize15:          dw      VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment15:      dw      VGAMEM_GRAPH
                        <1> WinBSegment15:      dw      0000h
                        <1> WinFuncPtr15:       dd      0
                        <1> BytesPerScanLine15: dw      2560
                        <1> XResolution15:      dw      640
                        <1> YResolution15:      dw      480
                        <1> XCharSize15:        db      8
                        <1> YCharSize15:        db      16
                        <1> NumberOfPlanes15:   db      1
                        <1> BitsPerPixel15:     db      32
                        <1> NumberOfBanks15:    db      19
                        <1> MemoryModel15:      db      VBE_MEMORYMODEL_DIRECT_COLOR
                        <1> BankSize15:         db      0
                        <1> NumberOfImagePages15:       db      12
                        <1> Reserved_page15:            db      0
                        <1> RedMaskSize15:              db      8
                        <1> RedFieldPosition15:         db      16
                        <1> GreenMaskSize15:            db      8
                        <1> GreenFieldPosition15:       db      8
                        <1> BlueMaskSize15:             db      8
                        <1> BlueFieldPosition15:        db      0
                        <1> RsvdMaskSize15:             db      8
                        <1> RsvdFieldPosition15:        db      24
                        <1> DirectColorModeInfo15:      db      VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE,
                        <1> PhysBasePtr15:              dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS,
                        <1> OffScreenMemOffset15:       dd      0
                        <1> OffScreenMemSize15:         dw      0
                        <1> LinBytesPerScanLine15:      dw      2560
                        <1> BnkNumberOfPages15:         db      0
                        <1> LinNumberOfPages15:         db      0
                        <1> LinRedMaskSize15:           db      8
                        <1> LinRedFieldPosition15:      db      16
                        <1> LinGreenMaskSize15:         db      8
                        <1> LinGreenFieldPosition15:    db      8
                        <1> LinBlueMaskSize15:          db      8
                        <1> LinBlueFieldPosition15:     db      0
                        <1> LinRsvdMaskSize15:          db      8
                        <1> LinRsvdFieldPosition15:     db      24
                        <1> MaxPixelClock15:            dd      0
                        <1>
                        <1>                     dw      0143h ; 800x600x32
                        <1> ModeAttributes16:   dw      MODE_ATTRIBUTES
                        <1> WinAAttributes16:   db      WINA_ATTRIBUTES
                        <1> WinBAttributes16:   db      0
                        <1> WinGranularity16:   dw      VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize16:          dw      VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment16:      dw      VGAMEM_GRAPH
                        <1> WinBSegment16:      dw      0000h
                        <1> WinFuncPtr16:       dd      0
                        <1> BytesPerScanLine16: dw      3200
                        <1> XResolution16:      dw      800
                        <1> YResolution16:      dw      600
                        <1> XCharSize16:        db      8
                        <1> YCharSize16:        db      16
                        <1> NumberOfPlanes16:   db      1
                        <1> BitsPerPixel16:     db      32
                        <1> NumberOfBanks16:    db      30
                        <1> MemoryModel16:      db      VBE_MEMORYMODEL_DIRECT_COLOR
                        <1> BankSize16:         db      0
                        <1> NumberOfImagePages16:       db      7
                        <1> Reserved_page16:            db      0
                        <1> RedMaskSize16:              db      8
                        <1> RedFieldPosition16:         db      16
                        <1> GreenMaskSize16:            db      8
                        <1> GreenFieldPosition16:       db      8
                        <1> BlueMaskSize16:             db      8
                        <1> BlueFieldPosition16:        db      0
                        <1> RsvdMaskSize16:             db      8
                        <1> RsvdFieldPosition16:        db      24
                        <1> DirectColorModeInfo16:      db      VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE,
                        <1> PhysBasePtr16:              dd      VBE_DISPI_LFB_PHYSICAL_ADDRESS,
                        <1> OffScreenMemOffset16:       dd      0
                        <1> OffScreenMemSize16:         dw      0
                        <1> LinBytesPerScanLine16:      dw      3200
                        <1> BnkNumberOfPages16:         db      0
                        <1> LinNumberOfPages16:         db      0
                        <1> LinRedMaskSize16:           db      8
                        <1> LinRedFieldPosition16:      db      16
                        <1> LinGreenMaskSize16:         db      8
                        <1> LinGreenFieldPosition16:db      8
                        <1> LinBlueMaskSize16:          db      8
                        <1> LinBlueFieldPosition16:     db      0
                        <1> LinRsvdMaskSize16:          db      8
                        <1> LinRsvdFieldPosition16:     db      24
                        <1> MaxPixelClock16:            dd      0
                        <1>
                        <1>                     dw      0144h ; 1024x768x32
                        <1> ModeAttributes17:   dw      MODE_ATTRIBUTES
                        <1> WinAAttributes17:   db      WINA_ATTRIBUTES
                        <1> WinBAttributes17:   db      0
                        <1> WinGranularity17:   dw      VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize17:          dw      VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment17:      dw      VGAMEM_GRAPH
                        <1> WinBSegment17:      dw      0000h
                        <1> WinFuncPtr17:       dd      0
                        <1> BytesPerScanLine17: dw      4096
                        <1> XResolution17:      dw      1024
                        <1> YResolution17:      dw      768
```

```
4269                            <1> XCharSize17:       db     8
4270                            <1> YCharSize17:       db     16
4271                            <1> NumberOfPlanes17:  db     1
4272                            <1> BitsPerPixel17:    db     32
4273                            <1> NumberOfBanks17:   db     48
4274                            <1> MemoryModel17:     db     VBE_MEMORYMODEL_DIRECT_COLOR
4275                            <1> BankSize17:        db     0
4276                            <1> NumberOfImagePages17:     db     4
4277                            <1> Reserved_page17:          db     0
4278                            <1> RedMaskSize17:            db     8
4279                            <1> RedFieldPosition17:       db     16
4280                            <1> GreenMaskSize17:          db     8
4281                            <1> GreenFieldPosition17:     db     8
4282                            <1> BlueMaskSize17:           db     8
4283                            <1> BlueFieldPosition17:      db     0
4284                            <1> RsvdMaskSize17:           db     8
4285                            <1> RsvdFieldPosition17:      db     24
4286                            <1> DirectColorModeInfo17:    db     VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
4287                            <1> PhysBasePtr17:            dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
4288                            <1> OffScreenMemOffset17:     dd     0
4289                            <1> OffScreenMemSize17:       dw     0
4290                            <1> LinBytesPerScanLine17:    dw     4096
4291                            <1> BnkNumberOfPages17:       db     0
4292                            <1> LinNumberOfPages17:       db     0
4293                            <1> LinRedMaskSize17:         db     8
4294                            <1> LinRedFieldPosition17:    db     16
4295                            <1> LinGreenMaskSize17:       db     8
4296                            <1> LinGreenFieldPosition17:db       8
4297                            <1> LinBlueMaskSize17:        db     8
4298                            <1> LinBlueFieldPosition17:   db     0
4299                            <1> LinRsvdMaskSize17:        db     8
4300                            <1> LinRsvdFieldPosition17:   db     24
4301                            <1> MaxPixelClock17:          dd     0
4302                            <1>
4303                            <1>                    dw     0146h ; 320x200x8
4304                            <1> ModeAttributes18:  dw     MODE_ATTRIBUTES
4305                            <1> WinAAttributes18:  db     WINA_ATTRIBUTES
4306                            <1> WinBAttributes18:  db     0
4307                            <1> WinGranularity18:  dw     VBE_DISPI_BANK_SIZE_KB
4308                            <1> WinSize18:         dw     VBE_DISPI_BANK_SIZE_KB
4309                            <1> WinASegment18:     dw     VGAMEM_GRAPH
4310                            <1> WinBSegment18:     dw     0000h
4311                            <1> WinFuncPtr18:      dd     0
4312                            <1> BytesPerScanLine18: dw    320
4313                            <1> XResolution18:     dw     320
4314                            <1> YResolution18:     dw     200
4315                            <1> XCharSize18:       db     8
4316                            <1> YCharSize18:       db     16
4317                            <1> NumberOfPlanes18:  db     1
4318                            <1> BitsPerPixel18:    db     8
4319                            <1> NumberOfBanks18:   db     1
4320                            <1> MemoryModel18:     db     VBE_MEMORYMODEL_PACKED_PIXEL
4321                            <1> BankSize18:        db     0
4322                            <1> NumberOfImagePages18:     db     255 ; 261 in vbetables.h (03/01/2020) !
4323                            <1> Reserved_page18:          db     0
4324                            <1> RedMaskSize18:            db     0
4325                            <1> RedFieldPosition18:       db     0
4326                            <1> GreenMaskSize18:          db     0
4327                            <1> GreenFieldPosition18:     db     0
4328                            <1> BlueMaskSize18:           db     0
4329                            <1> BlueFieldPosition18:      db     0
4330                            <1> RsvdMaskSize18:           db     0
4331                            <1> RsvdFieldPosition18:      db     0
4332                            <1> DirectColorModeInfo18:    db     0
4333                            <1> PhysBasePtr18:            dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
4334                            <1> OffScreenMemOffset18:     dd     0
4335                            <1> OffScreenMemSize18:       dw     0
4336                            <1> LinBytesPerScanLine18:    dw     320
4337                            <1> BnkNumberOfPages18:       db     0
4338                            <1> LinNumberOfPages18:       db     0
4339                            <1> LinRedMaskSize18:         db     0
4340                            <1> LinRedFieldPosition18:    db     0
4341                            <1> LinGreenMaskSize18:       db     0
4342                            <1> LinGreenFieldPosition18:  db     0
4343                            <1> LinBlueMaskSize18:        db     0
4344                            <1> LinBlueFieldPosition18:   db     0
4345                            <1> LinRsvdMaskSize18:        db     0
4346                            <1> LinRsvdFieldPosition18:   db     0
4347                            <1> MaxPixelClock18:          dd     0
4348                            <1>
4349                            <1>                    dw     018Dh ; 1280x720x16
4350                            <1> ModeAttributes19:  dw     MODE_ATTRIBUTES
4351                            <1> WinAAttributes19:  db     WINA_ATTRIBUTES
4352                            <1> WinBAttributes19:  db     0
4353                            <1> WinGranularity19:  dw     VBE_DISPI_BANK_SIZE_KB
4354                            <1> WinSize19:         dw     VBE_DISPI_BANK_SIZE_KB
4355                            <1> WinASegment19:     dw     VGAMEM_GRAPH
4356                            <1> WinBSegment19:     dw     0000h
4357                            <1> WinFuncPtr19:      dd     0
4358                            <1> BytesPerScanLine19: dw    2560
4359                            <1> XResolution19:     dw     1280
4360                            <1> YResolution19:     dw     720
4361                            <1> XCharSize19:       db     8
4362                            <1> YCharSize19:       db     16
4363                            <1> NumberOfPlanes19:  db     1
4364                            <1> BitsPerPixel19:    db     16
4365                            <1> NumberOfBanks19:   db     29
4366                            <1> MemoryModel19:     db     VBE_MEMORYMODEL_DIRECT_COLOR
4367                            <1> BankSize19:        db     0
4368                            <1> NumberOfImagePages19:     db     8
4369                            <1> Reserved_page19:          db     0
4370                            <1> RedMaskSize19:            db     5
4371                            <1> RedFieldPosition19:       db     11
4372                            <1> GreenMaskSize19:          db     6
4373                            <1> GreenFieldPosition19:     db     5
```

```
4374                                    <1> BlueMaskSize19:           db     5
4375                                    <1> BlueFieldPosition19:      db     0
4376                                    <1> RsvdMaskSize19:           db     0
4377                                    <1> RsvdFieldPosition19:      db     0
4378                                    <1> DirectColorModeInfo19:    db     0
4379                                    <1> PhysBasePtr19:            dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
4380                                    <1> OffScreenMemOffset19:     dd     0
4381                                    <1> OffScreenMemSize19:       dw     0
4382                                    <1> LinBytesPerScanLine19:    dw     2560
4383                                    <1> BnkNumberOfPages19:       db     0
4384                                    <1> LinNumberOfPages19:       db     0
4385                                    <1> LinRedMaskSize19:         db     5
4386                                    <1> LinRedFieldPosition19:    db     11
4387                                    <1> LinGreenMaskSize19:       db     6
4388                                    <1> LinGreenFieldPosition19:db       5
4389                                    <1> LinBlueMaskSize19:        db     5
4390                                    <1> LinBlueFieldPosition19:   db     0
4391                                    <1> LinRsvdMaskSize19:        db     0
4392                                    <1> LinRsvdFieldPosition19:   db     0
4393                                    <1> MaxPixelClock19:          dd     0
4394                                    <1>
4395                                    <1>                   dw     018Eh ; 1280x720x24
4396                                    <1> ModeAttributes20:  dw     MODE_ATTRIBUTES
4397                                    <1> WinAAttributes20:  db     WINA_ATTRIBUTES
4398                                    <1> WinBAttributes20:  db     0
4399                                    <1> WinGranularity20:  dw     VBE_DISPI_BANK_SIZE_KB
4400                                    <1> WinSize20:         dw     VBE_DISPI_BANK_SIZE_KB
4401                                    <1> WinASegment20:     dw     VGAMEM_GRAPH
4402                                    <1> WinBSegment20:     dw     0000h
4403                                    <1> WinFuncPtr20:      dd     0
4404                                    <1> BytesPerScanLine20: dw    3840
4405                                    <1> XResolution20:     dw     1280
4406                                    <1> YResolution20:     dw     720
4407                                    <1> XCharSize20:       db     8
4408                                    <1> YCharSize20:       db     16
4409                                    <1> NumberOfPlanes20:  db     1
4410                                    <1> BitsPerPixel20:    db     24
4411                                    <1> NumberOfBanks20:   db     43
4412                                    <1> MemoryModel20:     db     VBE_MEMORYMODEL_DIRECT_COLOR
4413                                    <1> BankSize20:        db     0
4414                                    <1> NumberOfImagePages20:     db     5
4415                                    <1> Reserved_page20:          db     0
4416                                    <1> RedMaskSize20:            db     8
4417                                    <1> RedFieldPosition20:       db     16
4418                                    <1> GreenMaskSize20:          db     8
4419                                    <1> GreenFieldPosition20:     db     8
4420                                    <1> BlueMaskSize20:           db     8
4421                                    <1> BlueFieldPosition20:      db     0
4422                                    <1> RsvdMaskSize20:           db     0
4423                                    <1> RsvdFieldPosition20:      db     0
4424                                    <1> DirectColorModeInfo20:    db     0
4425                                    <1> PhysBasePtr20:            dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
4426                                    <1> OffScreenMemOffset20:     dd     0
4427                                    <1> OffScreenMemSize20:       dw     0
4428                                    <1> LinBytesPerScanLine20:    dw     3840
4429                                    <1> BnkNumberOfPages20:       db     0
4430                                    <1> LinNumberOfPages20:       db     0
4431                                    <1> LinRedMaskSize20:         db     8
4432                                    <1> LinRedFieldPosition20:    db     16
4433                                    <1> LinGreenMaskSize20:       db     8
4434                                    <1> LinGreenFieldPosition20:  db     8
4435                                    <1> LinBlueMaskSize20:        db     8
4436                                    <1> LinBlueFieldPosition20:   db     0
4437                                    <1> LinRsvdMaskSize20:        db     0
4438                                    <1> LinRsvdFieldPosition20:   db     0
4439                                    <1> MaxPixelClock20:          dd     0
4440                                    <1>
4441                                    <1>                   dw     018Fh ; 1280x720x32
4442                                    <1> ModeAttributes21:  dw     MODE_ATTRIBUTES
4443                                    <1> WinAAttributes21:  db     WINA_ATTRIBUTES
4444                                    <1> WinBAttributes21:  db     0
4445                                    <1> WinGranularity21:  dw     VBE_DISPI_BANK_SIZE_KB
4446                                    <1> WinSize21:         dw     VBE_DISPI_BANK_SIZE_KB
4447                                    <1> WinASegment21:     dw     VGAMEM_GRAPH
4448                                    <1> WinBSegment21:     dw     0000h
4449                                    <1> WinFuncPtr21:      dd     0
4450                                    <1> BytesPerScanLine21: dw    5120
4451                                    <1> XResolution21:     dw     1280
4452                                    <1> YResolution21:     dw     720
4453                                    <1> XCharSize21:       db     8
4454                                    <1> YCharSize21:       db     16
4455                                    <1> NumberOfPlanes21:  db     1
4456                                    <1> BitsPerPixel21:    db     32
4457                                    <1> NumberOfBanks21:   db     57
4458                                    <1> MemoryModel21:     db     VBE_MEMORYMODEL_DIRECT_COLOR
4459                                    <1> BankSize21:        db     0
4460                                    <1> NumberOfImagePages21:     db     3
4461                                    <1> Reserved_page21:          db     0
4462                                    <1> RedMaskSize21:            db     8
4463                                    <1> RedFieldPosition21:       db     16
4464                                    <1> GreenMaskSize21:          db     8
4465                                    <1> GreenFieldPosition21:     db     8
4466                                    <1> BlueMaskSize21:           db     8
4467                                    <1> BlueFieldPosition21:      db     0
4468                                    <1> RsvdMaskSize21:           db     8
4469                                    <1> RsvdFieldPosition21:      db     24
4470                                    <1> DirectColorModeInfo21:    db     VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
4471                                    <1> PhysBasePtr21:            dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
4472                                    <1> OffScreenMemOffset21:     dd     0
4473                                    <1> OffScreenMemSize21:       dw     0
4474                                    <1> LinBytesPerScanLine21:    dw     5120
4475                                    <1> BnkNumberOfPages21:       db     0
4476                                    <1> LinNumberOfPages21:       db     0
4477                                    <1> LinRedMaskSize21:         db     8
4478                                    <1> LinRedFieldPosition21:    db     16
```

```
4479                            <1> LinGreenMaskSize21:       db    8
4480                            <1> LinGreenFieldPosition21:  db    8
4481                            <1> LinBlueMaskSize21:        db    8
4482                            <1> LinBlueFieldPosition21:   db    0
4483                            <1> LinRsvdMaskSize21:        db    8
4484                            <1> LinRsvdFieldPosition21:   db    24
4485                            <1> MaxPixelClock21:          dd    0
4486                            <1>
4487                            <1>                   dw    0190h ; 1920x1080x16
4488                            <1> ModeAttributes22: dw    MODE_ATTRIBUTES
4489                            <1> WinAAttributes22: db    WINA_ATTRIBUTES
4490                            <1> WinBAttributes22: db    0
4491                            <1> WinGranularity22: dw    VBE_DISPI_BANK_SIZE_KB
4492                            <1> WinSize22:        dw    VBE_DISPI_BANK_SIZE_KB
4493                            <1> WinASegment22:    dw    VGAMEM_GRAPH
4494                            <1> WinBSegment22:    dw    0000h
4495                            <1> WinFuncPtr22:     dd    0
4496                            <1> BytesPerScanLine22: dw  3840
4497                            <1> XResolution22:    dw    1920
4498                            <1> YResolution22:    dw    1080
4499                            <1> XCharSize22:      db    8
4500                            <1> YCharSize22:      db    16
4501                            <1> NumberOfPlanes22: db    1
4502                            <1> BitsPerPixel22:   db    16
4503                            <1> NumberOfBanks22:  db    64
4504                            <1> MemoryModel22:    db    VBE_MEMORYMODEL_DIRECT_COLOR,
4505                            <1> BankSize22:       db    0
4506                            <1> NumberOfImagePages22:   db    3
4507                            <1> Reserved_page22:        db    0
4508                            <1> RedMaskSize22:          db    5
4509                            <1> RedFieldPosition22:     db    11
4510                            <1> GreenMaskSize22:        db    6
4511                            <1> GreenFieldPosition22:   db    5
4512                            <1> BlueMaskSize22:         db    5
4513                            <1> BlueFieldPosition22:    db    0
4514                            <1> RsvdMaskSize22:         db    0
4515                            <1> RsvdFieldPosition22:    db    0
4516                            <1> DirectColorModeInfo22:  db    0
4517                            <1> PhysBasePtr22:          dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
4518                            <1> OffScreenMemOffset22:   dd    0
4519                            <1> OffScreenMemSize22:     dw    0
4520                            <1> LinBytesPerScanLine22:  dw    3840
4521                            <1> BnkNumberOfPages22:     db    0
4522                            <1> LinNumberOfPages22:     db    0
4523                            <1> LinRedMaskSize22:       db    5
4524                            <1> LinRedFieldPosition22:  db    11
4525                            <1> LinGreenMaskSize22:     db    6
4526                            <1> LinGreenFieldPosition22: db   5
4527                            <1> LinBlueMaskSize22:      db    5
4528                            <1> LinBlueFieldPosition22: db    0
4529                            <1> LinRsvdMaskSize22:      db    0
4530                            <1> LinRsvdFieldPosition22: db    0
4531                            <1> MaxPixelClock22:        dd    0
4532                            <1>
4533                            <1>                   dw    0191h ; 1920x1080x24
4534                            <1> ModeAttributes23: dw    MODE_ATTRIBUTES
4535                            <1> WinAAttributes23: db    WINA_ATTRIBUTES
4536                            <1> WinBAttributes23: db    0
4537                            <1> WinGranularity23: dw    VBE_DISPI_BANK_SIZE_KB
4538                            <1> WinSize23:        dw    VBE_DISPI_BANK_SIZE_KB
4539                            <1> WinASegment23:    dw    VGAMEM_GRAPH
4540                            <1> WinBSegment23:    dw    0000h
4541                            <1> WinFuncPtr23:     dd    0
4542                            <1> BytesPerScanLine23: dw  5760
4543                            <1> XResolution23:    dw    1920
4544                            <1> YResolution23:    dw    1080
4545                            <1> XCharSize23:      db    8
4546                            <1> YCharSize23:      db    16
4547                            <1> NumberOfPlanes23: db    1
4548                            <1> BitsPerPixel23:   db    24
4549                            <1> NumberOfBanks23:  db    95
4550                            <1> MemoryModel23:    db    VBE_MEMORYMODEL_DIRECT_COLOR
4551                            <1> BankSize23:       db    0
4552                            <1> NumberOfImagePages23:   db    1
4553                            <1> Reserved_page23:        db    0
4554                            <1> RedMaskSize23:          db    8
4555                            <1> RedFieldPosition23:     db    16
4556                            <1> GreenMaskSize23:        db    8
4557                            <1> GreenFieldPosition23:   db    8
4558                            <1> BlueMaskSize23:         db    8
4559                            <1> BlueFieldPosition23:    db    0
4560                            <1> RsvdMaskSize23:         db    0
4561                            <1> RsvdFieldPosition23:    db    0
4562                            <1> DirectColorModeInfo23:  db    0
4563                            <1> PhysBasePtr23:          dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
4564                            <1> OffScreenMemOffset23:   dd    0
4565                            <1> OffScreenMemSize23:     dw    0
4566                            <1> LinBytesPerScanLine23:  dw    5760
4567                            <1> BnkNumberOfPages23:     db    0
4568                            <1> LinNumberOfPages23:     db    0
4569                            <1> LinRedMaskSize23:       db    8
4570                            <1> LinRedFieldPosition23:  db    16
4571                            <1> LinGreenMaskSize23:     db    8
4572                            <1> LinGreenFieldPosition23:db    8
4573                            <1> LinBlueMaskSize23:      db    8
4574                            <1> LinBlueFieldPosition23: db    0
4575                            <1> LinRsvdMaskSize23:      db    0
4576                            <1> LinRsvdFieldPosition23: db    0
4577                            <1> MaxPixelClock23:        dd    0
4578                            <1>
4579                            <1>                   dw    0192h ; 1920x1080x32
4580                            <1> ModeAttributes24: dw    MODE_ATTRIBUTES
4581                            <1> WinAAttributes24: db    WINA_ATTRIBUTES
4582                            <1> WinBAttributes24: db    0
4583                            <1> WinGranularity24: dw    VBE_DISPI_BANK_SIZE_KB
```

```
4584                                    <1> WinSize24:          dw    VBE_DISPI_BANK_SIZE_KB
4585                                    <1> WinASegment24:      dw    VGAMEM_GRAPH
4586                                    <1> WinBSegment24:      dw    0000h
4587                                    <1> WinFuncPtr24:       dd    0
4588                                    <1> BytesPerScanLine24: dw    7680
4589                                    <1> XResolution24:      dw    1920
4590                                    <1> YResolution24:      dw    1080
4591                                    <1> XCharSize24:        db    8
4592                                    <1> YCharSize24:        db    16
4593                                    <1> NumberOfPlanes24:   db    1
4594                                    <1> BitsPerPixel24:     db    32
4595                                    <1> NumberOfBanks24:    db    127
4596                                    <1> MemoryModel24:      db    VBE_MEMORYMODEL_DIRECT_COLOR
4597                                    <1> BankSize24:         db    0
4598                                    <1> NumberOfImagePages24:     db    1
4599                                    <1> Reserved_page24:          db    0
4600                                    <1> RedMaskSize24:            db    8
4601                                    <1> RedFieldPosition24:       db    16
4602                                    <1> GreenMaskSize24:          db    8
4603                                    <1> GreenFieldPosition24:     db    8
4604                                    <1> BlueMaskSize24:           db    8
4605                                    <1> BlueFieldPosition24:      db    0
4606                                    <1> RsvdMaskSize24:           db    8
4607                                    <1> RsvdFieldPosition24:      db    24
4608                                    <1> DirectColorModeInfo24:    db    VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
4609                                    <1> PhysBasePtr24:            dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
4610                                    <1> OffScreenMemOffset24:     dd    0
4611                                    <1> OffScreenMemSize24:       dw    0
4612                                    <1> LinBytesPerScanLine24:    dw    7680
4613                                    <1> BnkNumberOfPages24:       db    0
4614                                    <1> LinNumberOfPages24:       db    0
4615                                    <1> LinRedMaskSize24:         db    8
4616                                    <1> LinRedFieldPosition24:    db    16
4617                                    <1> LinGreenMaskSize24:       db    8
4618                                    <1> LinGreenFieldPosition24:  db    8
4619                                    <1> LinBlueMaskSize24:        db    8
4620                                    <1> LinBlueFieldPosition24:   db    0
4621                                    <1> LinRsvdMaskSize24:        db    8
4622                                    <1> LinRsvdFieldPosition24:   db    24
4623                                    <1> MaxPixelClock24:          dd    0
4624                                    <1>
4625                                    <1> VBE_VESA_MODE_END_OF_LIST: dw   0
4626                                    <1>
4627                                    <1> %endif
4628                                    <1>
4629                                    <1> ; 24/11/2020
4630                                    <1>
4631                                    <1> direct_color_fields:
4632                                    <1>       ; 24/11/2020
4633                                    <1>
4634                                    <1> ; (vbetables-gen.c)
4635                                    <1> ; // Direct Color fields
4636                                    <1> ; (required for direct/6 and YUV/7 memory models)
4637                                    <1> ; switch(pm->depth) {
4638                                    <1>
4639                                    <1> ;case 8:
4640 00006F06 00                      <1> r_size_8:  db 0
4641 00006F07 00                      <1> r_pos_8:   db 0
4642 00006F08 00                      <1> g_size_8:  db 0
4643 00006F09 00                      <1> g_pos_8:   db 0
4644 00006F0A 00                      <1> b_size_8:  db 0
4645 00006F0B 00                      <1> b_pos_8:   db 0
4646 00006F0C 00                      <1> a_size_8:  db 0
4647 00006F0D 00                      <1> a_pos_8:   db 0
4648                                    <1>
4649                                    <1> ;case 16:
4650 00006F0E 05                      <1> r_size_16:  db 5
4651 00006F0F 0B                      <1> r_pos_16:   db 11
4652 00006F10 06                      <1> g_size_16:  db 6
4653 00006F11 05                      <1> g_pos_16:   db 5
4654 00006F12 05                      <1> b_size_16:  db 5
4655 00006F13 00                      <1> b_pos_16:   db 0
4656 00006F14 00                      <1> a_size_16:  db 0
4657 00006F15 00                      <1> a_pos_16:   db 0
4658                                    <1>
4659                                    <1> ;case 24:
4660 00006F16 08                      <1> r_size_24:  db 8
4661 00006F17 10                      <1> r_pos_24:   db 16
4662 00006F18 08                      <1> g_size_24:  db 8
4663 00006F19 08                      <1> g_pos_24:   db 8
4664 00006F1A 08                      <1> b_size_24:  db 8
4665 00006F1B 00                      <1> b_pos_24:   db 0
4666 00006F1C 00                      <1> a_size_24:  db 0
4667 00006F1D 00                      <1> a_pos_24:   db 0
4668                                    <1>
4669                                    <1> ;case 32:
4670 00006F1E 08                      <1> r_size_32:  db 8
4671 00006F1F 10                      <1> r_pos_32:   db 16
4672 00006F20 08                      <1> g_size_32:  db 8
4673 00006F21 08                      <1> g_pos_32:   db 8
4674 00006F22 08                      <1> b_size_32:  db 8
4675 00006F23 00                      <1> b_pos_32:   db 0
4676 00006F24 08                      <1> a_size_32:  db 8
4677 00006F25 18                      <1> a_pos_32:   db 24
3099                                        ;%include 'diskdata.s'    ; DISK (BIOS) DATA (initialized)
3100                                        ;;;
3101
3102                                        Align 2
3103
3104                                        %include 'sysdefs.s' ; 24/01/2015
3105                                    <1> ; ****************************************************************************
3106                                    <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - SYSTEM DEFINITIONS : sysdefs.s
3107                                    <1> ; ----------------------------------------------------------------------------
3108                                    <1> ; Last Update: 31/12/2017
3109                                    <1> ; ----------------------------------------------------------------------------
```

```
3110                            <1> ; Beginning: 24/01/2016
3111                            <1> ; ---------------------------------------------------------------------------
3112                            <1> ; Assembler: NASM version 2.11 (trdos386.s)
3113                            <1> ; ---------------------------------------------------------------------------
3114                            <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
3115                            <1> ; sysdefs.inc (14/11/2015)
3116                            <1> ; ***************************************************************************
3117                            <1>
3118                            <1> ; Retro UNIX 386 v1 Kernel - SYSDEFS.INC
3119                            <1> ; Last Modification: 14/11/2015
3120                            <1> ;
3121                            <1> ; ///////// RETRO UNIX 386 V1 SYSTEM DEFINITIONS ///////////////
3122                            <1> ; (Modified from
3123                            <1> ;     Retro UNIX 8086 v1 system definitions in 'UNIX.ASM', 01/09/2014)
3124                            <1> ; ((UNIX.ASM (RETRO UNIX 8086 V1 Kernel), 11/03/2013 - 01/09/2014))
3125                            <1> ;     UNIX.ASM (MASM 6.11) --> SYSDEFS.INC (NASM 2.11)
3126                            <1> ; ---------------------------------------------------------------------------
3127                            <1> ;
3128                            <1> ; Derived from UNIX Operating System (v1.0 for PDP-11)
3129                            <1> ; (Original) Source Code by Ken Thompson (1971-1972)
3130                            <1> ; <Bell Laboratories (17/3/1972)>
3131                            <1> ; <Preliminary Release of UNIX Implementation Document>
3132                            <1> ;
3133                            <1> ; ***************************************************************************
3134                            <1>
3135                            <1> nproc equ   16  ; number of processes
3136                            <1> nfiles equ  50
3137                            <1> ntty  equ   8   ; 8+1 -> 8 (10/05/2013)
3138                            <1> nbuf  equ   4   ; 6 ;; 21/08/2015 - 'namei' buffer problem when nbuf > 4
3139                            <1>             ; NOTE: If fd0 super block buffer addres is beyond of the 1st
3140                            <1>             ; 32K, DMA r/w routine or someting else causes a jump to
3141                            <1>             ; kernel panic routine (in 'alloc' routine, in u5.s)
3142                            <1>             ; because of invalid buffer content (r/w error).
3143                            <1>             ; When all buffers are set before the end of the 1st 32k,
3144                            <1>             ; there is no problem!? (14/11/2015)
3145                            <1>
3146                            <1> ;csgmnt equ 2000h  ; 26/05/2013 (segment of process 1)
3147                            <1> ;core equ   0          ; 19/04/2013
3148                            <1> ;ecore equ  32768 - 64  ; 04/06/2013 (24/05/2013)
3149                            <1>       ; (if total size of argument list and arguments is 128 bytes)
3150                            <1>       ; maximum executable file size = 32768-(64+40+128-6) = 32530 bytes
3151                            <1>       ; maximum stack size = 40 bytes (+6 bytes for 'IRET' at 32570)
3152                            <1>       ; initial value of user's stack pointer = 32768-64-128-2 = 32574
3153                            <1>       ;     (sp=32768-args_space-2 at the beginning of execution)
3154                            <1>       ; argument list offset = 32768-64-128 = 32576 (if it is 128 bytes)
3155                            <1>       ; 'u' structure offset (for the '/core' dump file) = 32704
3156                            <1>       ; '/core' dump file size = 32768 bytes
3157                            <1>
3158                            <1> ; 08/03/2014
3159                            <1> ;sdsegmnt equ 6C0h  ; 256*16 bytes (swap data segment size for 16 processes)
3160                            <1> ; 19/04/2013 Retro UNIX 8086 v1 feaure only !
3161                            <1> ;;sdsegmnt equ 740h  ; swap data segment (for user structures and registers)
3162                            <1>
3163                            <1> ; 30/08/2013
3164                            <1> time_count equ 4 ; 10 --> 4 01/02/2014
3165                            <1>
3166                            <1> ; 05/02/2014
3167                            <1> ; process status
3168                            <1> ;SFREE      equ 0
3169                            <1> ;SRUN equ 1
3170                            <1> ;SWAIT      equ 2
3171                            <1> ;SZOMB      equ 3
3172                            <1> ;SSLEEP     equ 4 ; Retro UNIX 8086 V1 extension (for sleep and wakeup)
3173                            <1>
3174                            <1> ; 09/03/2015
3175                            <1> userdata equ 80000h ; user structure data address for current user ; temporary
3176                            <1> swap_queue equ 90000h - 2000h ; swap queue address ; temporary
3177                            <1> swap_alloc_table equ 0D0000h  ;  swap allocation table address ; temporary
3178                            <1>
3179                            <1> ; 17/09/2015
3180                            <1> ESPACE equ 48 ; [u.usp] (at 'sysent') - [u.sp] value for error return
3181                            <1>
3182                            <1> ; 31/12/2017
3183                            <1> ; 19/02/2017
3184                            <1> ; 15/10/2016
3185                            <1> ; 20/05/2016
3186                            <1> ; 19/05/2016
3187                            <1> ; 18/05/2016
3188                            <1> ; 29/04/2016
3189                            <1> ; TRDOS 386 (TRDOS v2.0) system calls - temporary List
3190                            <1> ; 14/07/2013 - 21/09/2015 (Retro UNIX 8086 & 386 system calls)
3191                            <1> _ver        equ 0 ; Get TRDOS version (v2.0)
3192                            <1> _exit       equ 1
3193                            <1> _fork       equ 2
3194                            <1> _read       equ 3
3195                            <1> _write      equ 4
3196                            <1> _open       equ 5
3197                            <1> _close      equ 6
3198                            <1> _wait       equ 7
3199                            <1> _creat      equ 8
3200                            <1> _rename     equ 9  ; TRDOS 386, Rename File (31/12/2017)
3201                            <1> _delete     equ 10 ; TRDOS 386, Delete File (29/12/2017)
3202                            <1> _exec       equ 11
3203                            <1> _chdir      equ 12
3204                            <1> _time       equ 13 ; TRDOS 386, Get Sys Date&Time (30/12/2017)
3205                            <1> _mkdir      equ 14
3206                            <1> _chmod      equ 15 ; TRDOS 386, Change Attributes (30/12/2017)
3207                            <1> _rmdir      equ 16 ; TRDOS 386, Remove Directory (29/12/2017)
3208                            <1> _break      equ 17
3209                            <1> _drive      equ 18 ; TRDOS 386, Get/Set Current Drv (30/12/2017)
3210                            <1> _seek       equ 19
3211                            <1> _tell       equ 20
3212                            <1> _mem        equ 21 ; TRDOS 386, Get Total&Free Mem (31/12/2017)
3213                            <1> _prompt     equ 22 ; TRDOS 386, Change Cmd Prompt (31/12/2017)
3214                            <1> _path       equ 23 ; TRDOS 386, Get/Set Run Path (31/12/2017)
```

```
3215                    <1> _env        equ 24 ; TRDOS 386, Get/Set Env Vars (31/12/2017)
3216                    <1> _stime      equ 25 ; TRDOS 386, Set Sys Date&Time (30/12/2017)
3217                    <1> _quit       equ 26
3218                    <1> _intr       equ 27
3219                    <1> _dir        equ 28 ; TRDOS 386, Get Curr Drive&Dir (30/12/2017)
3220                    <1> _emt        equ 29
3221                    <1> _ldrvt      equ 30 ; TRDOS 386, Get Logical DOS DDT (30/12/2017)
3222                    <1> _video      equ 31 ; TRDOS 386 Video Functions (16/05/2016)
3223                    <1> _audio      equ 32 ; TRDOS 386 Video Functions (16/05/2016)
3224                    <1> _timer      equ 33 ; TRDOS 386 Timer Functions (18/05/2016)
3225                    <1> _sleep      equ 34 ; Retro UNIX 8086 v1 feature only !
3226                    <1> _msg        equ 35 ; Retro UNIX 386 v1 feature only !
3227                    <1> _geterr     equ 36 ; Retro UNIX 386 v1 feature only !
3228                    <1> _fpsave     equ 37 ; TRDOS 386 FPU state option (28/02/2017)
3229                    <1> _pri        equ 38 ; change priority - TRDOS 386 (20/05/2016)
3230                    <1> _rele       equ 39 ; TRDOS 386 (19/05/2016)
3231                    <1> _fff        equ 40 ; Find First File - TRDOS 386 (15/10/2016)
3232                    <1> _fnf        equ 41 ; Find Next File - TRDOS 386 (15/10/2016)
3233                    <1> _alloc      equ 42 ; Allocate memory - TRDOS 386 (19/02/2017)
3234                    <1>             ; TRDOS 386 (19/02/2017) DMA buff fuctions
3235                    <1> _dalloc     equ 43 ; Deallocate mem - TRDOS 386 (19/02/2017)
3236                    <1> _calbac     equ 44 ; Set IRQ callback - TRDOS 386 (20/02/2017)
3237                    <1> _dma        equ 45 ; DMA service - TRDOS 386 (20/08/2017)
3238                    <1>
3239                    <1> %macro sys 1-4
3240                    <1>     ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
3241                    <1>     ; 03/09/2015
3242                    <1>     ; 13/04/2015
3243                    <1>     ; Retro UNIX 386 v1 system call.
3244                    <1>     %if %0 >= 2
3245                    <1>         mov ebx, %2
3246                    <1>         %if %0 >= 3
3247                    <1>             mov ecx, %3
3248                    <1>             %if %0 = 4
3249                    <1>                mov edx, %4
3250                    <1>             %endif
3251                    <1>         %endif
3252                    <1>     %endif
3253                    <1>     mov eax, %1
3254                    <1>     ;int 30h
3255                    <1>     int 40h ; TRDOS 386 (TRDOS v2.0)
3256                    <1> %endmacro
3257                    <1>
3258                    <1> ; TRDOS 386 system calls, interrupt number
3259                    <1> ; 25/12/2016
3260                    <1> SYSCALL_INT_NUM    equ '40' ; '40h'
3261                    <1>
3262                    <1> ; 13/05/2015 - ERROR CODES
3263                    <1> ERR_FILE_NOT_OPEN  equ 10 ; 'file not open !' error
3264                    <1> ERR_FILE_ACCESS    equ 11 ; 'permission denied !' error
3265                    <1> ; 14/05/2015
3266                    <1> ERR_DIR_ACCESS     equ 11 ; 'permission denied !' error
3267                    <1> ERR_FILE_NOT_FOUND equ 12 ; 'file not found !' error
3268                    <1> ERR_TOO_MANY_FILES equ 13 ; 'too many open files !' error
3269                    <1> ERR_DIR_EXISTS     equ 14 ; 'directory already exists !' error
3270                    <1> ; 16/05/2015
3271                    <1> ERR_DRV_NOT_RDY    equ 15 ; 'drive not ready !' error
3272                    <1> ; 18/05/2015
3273                    <1> ERR_DEV_NOT_RDY    equ 15 ; 'device not ready !' error
3274                    <1> ERR_DEV_ACCESS     equ 11 ; 'permission denied !' error
3275                    <1> ERR_DEV_NOT_OPEN   equ 10 ; 'device not open !' error
3276                    <1> ; 07/06/2015
3277                    <1> ERR_FILE_EOF       equ 16 ; 'end of file !' error
3278                    <1> ERR_DEV_VOL_SIZE   equ 16 ; 'out of volume !' error
3279                    <1> ; 09/06/2015
3280                    <1> ERR_DRV_READ       equ 17 ; 'disk read error !'
3281                    <1> ERR_DRV_WRITE      equ 18 ; 'disk write error !'
3282                    <1> ; 16/06/2015
3283                    <1> ERR_NOT_DIR        equ 19 ; 'not a (valid) directory !' error
3284                    <1> ERR_FILE_SIZE      equ 20 ; 'file size error !'
3285                    <1> ; 22/06/2015
3286                    <1> ERR_NOT_SUPERUSER  equ 11 ; 'permission denied !' error
3287                    <1> ERR_NOT_OWNER      equ 11 ; 'permission denied !' error
3288                    <1> ERR_NOT_FILE       equ 11 ; 'permission denied !' error
3289                    <1> ; 23/06/2015
3290                    <1> ERR_FILE_EXISTS    equ 14 ; 'file already exists !' error
3291                    <1> ERR_DRV_NOT_SAME   equ 21 ; 'not same drive !' error
3292                    <1> ERR_DIR_NOT_FOUND  equ 12 ; 'directory not found !' error
3293                    <1> ERR_NOT_EXECUTABLE equ 22 ; 'not executable file !' error
3294                    <1> ; 27/06/2015
3295                    <1> ERR_INV_PARAMETER  equ 23 ; 'invalid parameter !' error
3296                    <1> ERR_INV_DEV_NAME   equ 24 ; 'invalid device name !' error
3297                    <1> ; 29/06/2015
3298                    <1> ERR_TIME_OUT       equ 25 ; 'time out !' error
3299                    <1> ERR_DEV_NOT_RESP   equ 25 ; 'device not responding !' error
3300                    <1> ; 10/10/2016
3301                    <1> ERR_INV_FILE_NAME  equ 26 ; 'invalid file name !' error
3302                    <1> ERR_INV_FLAGS      equ 23 ; 'invalid flags !' error
3303                    <1> ; For code compatibility with previous version of TRDOS (2011)
3304                    <1> ; (Temporary error codes for current TRDOS 386 -2016- version)
3305                    <1> ERR_NO_MORE_FILES  equ 12 ; 'no more files !' error
3306                    <1> ERR_PATH_NOT_FOUND equ  3 ; 'path not found !' error
3307                    <1>                       ; 'dir not found !' ; TRDOS 8086
3308                    <1> ERR_NOT_FOUND      equ  2 ; 'file not found !' ; TRDOS 8086
3309                    <1> ERR_DISK_SPACE     equ 39 ; 'out of volume !' TRDOS 8086
3310                    <1>                       ; 'insufficient disk space !' ; 27h
3311                    <1> ERR_DISK_WRITE     equ 30 ; 'disk write protected !' ; 16/10/2016
3312                    <1> ERR_ACCESS_DENIED  equ  5 ; 'access denied !' ; TRDOS 8086
3313                    <1> ; 28/02/2017
3314                    <1> ERR_PERM_DENIED    equ 11 ; 'permission denied !' error
3315                    <1> ; 18/05/2016
3316                    <1> ERR_MISC           equ 27 ; miscellaneous/other errors
3317                    <1> ; 15/10/2016
3318                    <1> ; TRDOS 8086 -> TRDOS 386 (0Bh -> 28)
3319                    <1> ERR_INV_FORMAT     equ 28 ; 'invalid format !' error
```

```
                        <1> ; TRDOS 8086 -> TRDOS 386 (0Dh -> 29)
                        <1> ERR_INV_DATA       equ 29 ; 'invalid data !' error
                        <1> ; TRDOS 8086 -> TRDOS 386 (0Eh -> 20)
                        <1> ERR_ZERO_LENGTH    equ 20  ; 'zero length !' error
                        <1> ; TRDOS 8086 -> TRDOS 386 (15h -> 17, 1Dh -> 18, 1Eh -> 17)
                        <1> ERR_DRV_NR_READ    equ 17 ; 'drive not ready or read error !'
                        <1> ERR_DRV_NR_WRITE   equ 18 ; 'drive not ready or write error !'
                        <1> ; 15/10/2016
                        <1> ERR_INV_PATH_NAME  equ 19 ; 'bad path name !' error
                        <1> ERR_BAD_CMD_ARG    equ  1 ; 'bad command argument !' ; TRDOS 8086
                        <1> ERR_INV_FNUMBER    equ  1 ; 'invalid function number !' ; TRDOS 8086
                        <1> ERR_BIG_FILE       equ  8 ; 'big file & out of memory ! ; TRDOS 8086
                        <1> ERR_BIG_DATA       equ  8 ; 'big data & out of memory ! ; TRDOS 8086
                        <1> ERR_CLUSTER        equ 35 ; 'cluster not available !' ; TRDOS 8086
                        <1> ERR_OUT_OF_MEMORY  equ  4 ; 'out of memory !'
                        <1>                           ; 'insufficient memory !'
                        <1> ERR_P_VIOLATION    equ 6  ; 'protection violation !'
                        <1> ERR_PAGE_FAULT     equ 224 ;'page fault !' ;0E0h
                        <1> ERR_SWP_DISK_READ        equ 40
                        <1> ERR_SWP_DISK_NOT_PRESENT  equ 41
                        <1> ERR_SWP_SECTOR_NOT_PRESENT equ 42
                        <1> ERR_SWP_NO_FREE_SPACE     equ 43
                        <1> ERR_SWP_DISK_WRITE        equ 44
                        <1> ERR_SWP_NO_PAGE_TO_SWAP   equ 45
                        <1> ; 10/04/2017
                        <1> ERR_BUFFER     equ 46  ; 'buffer error !'
                        <1> ; 28/08/2017 (20/08/2017)
                        <1> ERR_DMA        equ -1  ; DMA buffer (allocation/misc.) error!
                        <1>
                        <1> ; 26/08/2015
                        <1> ; 24/07/2015
                        <1> ; 24/06/2015
                        <1> MAX_ARG_LEN    equ 256 ; max. length of sys exec arguments
                        <1> ; 01/07/2015
                        <1> MAX_MSG_LEN    equ 255 ; max. msg length for 'sysmsg'
                        <1> ;
                        <1> ; 06/10/2016
                        <1> OPENFILES      equ 10  ; max. number of open files (system)
                        <1> ; 07/10/2016
                        <1> ;NUMOFDEVICES   equ 20  ; max. num of available devices (sys)
                        <1>
                            %include 'trdosk0.s' ; 04/01/2016
                        <1> ; ****************************************************************************
                        <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - DEFINITIONS : trdosk0.s
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Last Update: 29/02/2016
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Beginning: 04/01/2016
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Assembler: NASM version 2.11 (trdos386.s)
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
                        <1> ; TRDOS2.ASM (09/11/2011)
                        <1> ; ****************************************************************************
                        <1> ; TRDOS2.ASM (c) 2004-2011 Erdogan TAN [ 17/01/2004 ] Last Update: 09/11/2011
                        <1> ;
                        <1> ; Masterboot / Partition Table at Beginning+1BEh
                        <1> ptBootable       equ 0
                        <1> ptBeginHead      equ 1
                        <1> ptBeginSector    equ 2
                        <1> ptBeginCylinder  equ 3
                        <1> ptFileSystemID   equ 4
                        <1> ptEndHead        equ 5
                        <1> ptEndSector      equ 6
                        <1> ptEndCylinder    equ 7
                        <1> ptStartSector    equ 8
                        <1> ptSectors        equ 12
                        <1>
                        <1> ; Boot Sector Parameters at 7C00h
                        <1> DataArea1     equ -4
                        <1> DataArea2     equ -2
                        <1> BootStart     equ 0h
                        <1> OemName       equ 03h
                        <1> BytesPerSec   equ 0Bh
                        <1> SecPerClust   equ 0Dh
                        <1> ResSectors    equ 0Eh
                        <1> FATs          equ 10h
                        <1> RootDirEnts   equ 11h
                        <1> Sectors       equ 13h
                        <1> Media         equ 15h
                        <1> FATSecs       equ 16h
                        <1> SecPerTrack   equ 18h
                        <1> Heads         equ 1Ah
                        <1> Hidden1       equ 1Ch
                        <1> Hidden2       equ 1Eh
                        <1> HugeSec1      equ 20h
                        <1> HugeSec2      equ 22h
                        <1> DriveNumber   equ 24h
                        <1> Reserved1     equ 25h
                        <1> bootsignature equ 26h
                        <1> VolumeID      equ 27h
                        <1> VolumeLabel   equ 2Bh
                        <1> FileSysType   equ 36h
                        <1> Reserved2     equ 3Eh                       ; Starting cluster of P2000
                        <1>
                        <1> ; FAT32 BPB Structure
                        <1> FAT32_FAT_Size equ 36
                        <1> FAT32_RootFClust equ 44
                        <1> FAT32_FSInfoSec equ 48
                        <1> FAT32_DrvNum equ 64
                        <1> FAT32_BootSig equ 66
                        <1> FAT32_VolID equ 67
                        <1> FAT32_VolLab equ 71
                        <1> FAT32_FilSysType equ 82
                        <1>
```

```
3169                                    <1> ; BIOS Disk Parameters
3170                                    <1> DPDiskNumber  equ 0h
3171                                    <1> DPDType       equ 1h
3172                                    <1> DPReturn      equ 2h
3173                                    <1> DPHeads       equ 3h
3174                                    <1> DPCylinders   equ 4h
3175                                    <1> DPSecPerTrack equ 6h
3176                                    <1> DPDisks       equ 7h
3177                                    <1> DPTableOff    equ 8h
3178                                    <1> DPTableSeg    equ 0Ah
3179                                    <1> DPNumOfSecs   equ 0Ch
3180                                    <1>
3181                                    <1> ; BIOS INT 13h Extensions (LBA extensions)
3182                                    <1> ; Just After DP Data (DPDiskNumber+)
3183                                    <1> DAP_PacketSize equ 10h  ; If extensions present, this byte will be >=10h
3184                                    <1> DAP_Reserved1 equ 11h   ; Reserved Byte
3185                                    <1> DAP_NumOfBlocks equ 12h ; Value of this byte must be 0 to 127
3186                                    <1> DAP_Reserved2 equ 13h   ; Reserved Byte
3187                                    <1> DAP_Destination equ 14h ; Address of Transfer Buffer as SEGMENT:OFFSET
3188                                    <1> DAP_LBA_Address equ 18h ; LBA=(C1*H0+H1)*S0+S1-1
3189                                    <1>                         ; C1= Selected Cylinder Number
3190                                    <1>                         ; H0= Number Of Heads (Maximum Head Number + 1)
3191                                    <1>                         ; H1= Selected Head Number
3192                                    <1>                         ; S0= Maximum Sector Number
3193                                    <1>                         ; S1= Selected Sector Number
3194                                    <1>                         ; QUAD WORD
3195                                    <1> ; DAP_Flat_Destination equ 20h ; 64 bit address, if value in 4h is FFFF:FFFFh
3196                                    <1>                               ; QUAD WORD (Also, value in 0h must be 18h)
3197                                    <1>                               ; TR-DOS will not use 64 bit Flat Address
3198                                    <1>
3199                                    <1> ; INT 13h Function 48h "Get Enhanced Disk Drive Parameters"
3200                                    <1> ; Just After DP Data (DPDiskNumber+)
3201                                    <1> GetDParams_48h equ 20h ; Word. Data Length, must be 26 (1Ah) for short data.
3202                                    <1> GDP_48h_InfoFlag equ 22h ; Word
3203                                    <1> ; Bit 1 = 1 -> The geometry returned in bytes 4-15 is valid.
3204                                    <1> GDP_48h_NumOfPCyls equ 24h ; Double Word. Number physical cylinders.
3205                                    <1> GDP_48h_NumOfPHeads equ 28h ; Double Word. Number of physical heads.
3206                                    <1> GDP_48h_NumOfPSpT equ 2Ch ; Double word. Num of physical sectors per track.
3207                                    <1> GDP_48h_LBA_Sectors equ 30h ; 8 bytes. Number of physical/LBA sectors.
3208                                    <1> GDP_48h_BytesPerSec equ 38h ; Word. Number of bytes in a sector.
3209                                    <1>
3210                                    <1> ; TR-DOS Standalone Program Extensions to the DiskParams Block
3211                                    <1> ; Just After DP Data (DPDiskNumber+)
3212                                    <1> TRDP_CurrentSector equ 3Ah  ; DX:AX (LBA)
3213                                    <1> TRDP_SectorCount equ 3Eh    ; CX (or Counter)
3214                                    <1>
3215                                    <1>
3216                                    <1> ; DOS Logical Disks
3217                                    <1> LD_Name equ 0
3218                                    <1> LD_DiskType equ 1
3219                                    <1> LD_PhyDrvNo equ 2
3220                                    <1> LD_FATType equ 3
3221                                    <1> LD_FSType equ 4
3222                                    <1> LD_LBAYes equ 5
3223                                    <1> LD_BPB equ 6
3224                                    <1> LD_FATBegin equ 96
3225                                    <1> LD_ROOTBegin equ 100
3226                                    <1> LD_DATABegin equ 104
3227                                    <1> LD_StartSector equ 108
3228                                    <1> LD_TotalSectors equ 112
3229                                    <1> LD_FreeSectors equ 116
3230                                    <1> LD_Clusters equ 120
3231                                    <1> LD_PartitionEntry equ 124
3232                                    <1> LD_DParamEntry equ 125
3233                                    <1> LD_MediaChanged equ 126
3234                                    <1> LD_CDirLevel equ 127
3235                                    <1> LD_CurrentDirectory equ 128
3236                                    <1>
3237                                    <1> ; Singlix FS Extensions to DOS Logical Disks
3238                                    <1> ; 03/01/2010 (LD_BPB compatibility for CHS r/w)
3239                                    <1>
3240                                    <1> LD_FS_Name equ 0
3241                                    <1> LD_FS_DiskType equ 1
3242                                    <1> LD_FS_PhyDrvNo equ 2
3243                                    <1> LD_FS_FATType equ 3
3244                                    <1> LD_FS_FSType equ 4
3245                                    <1> LD_FS_LBAYes equ 5
3246                                    <1> LD_FS_BPB equ 6
3247                                    <1> LD_FS_MediaAttrib equ 6
3248                                    <1> LD_FS_VersionMajor equ 7
3249                                    <1> LD_FS_RootDirD equ 8
3250                                    <1> LD_FS_MATLocation equ 12
3251                                    <1> LD_FS_Reserved1 equ 16 ;1 reserved byte
3252                                    <1> LD_FS_BytesPerSec equ 17 ; LD_BPB + 0Bh
3253                                    <1> LD_FS_Reserved2 equ 19 ;2 reserved byte
3254                                    <1> LD_FS_DATLocation equ 20
3255                                    <1> LD_FS_DATSectors equ 24
3256                                    <1> LD_FS_Reserved3 equ 28 ;3 reserved word
3257                                    <1> LD_FS_SecPerTrack equ 30 ; LD_BPB + 18h
3258                                    <1> LD_FS_NumHeads equ 32    ; LD_BPB + 1Ah
3259                                    <1> LD_FS_UnDelDirD equ 34
3260                                    <1> LD_FS_Reserved4 equ 38 ;4 reserved word
3261                                    <1> LD_FS_VolumeSerial equ 40
3262                                    <1> LD_FS_VolumeName equ 44
3263                                    <1> LD_FS_BeginSector equ 108
3264                                    <1> LD_FS_VolumeSize equ 112
3265                                    <1> LD_FS_FreeSectors equ 116
3266                                    <1> LD_FS_FirstFreeSector equ 120
3267                                    <1> LD_FS_PartitionEntry equ 124
3268                                    <1> LD_FS_DParamEntry equ 125
3269                                    <1> LD_FS_MediaChanged equ 126
3270                                    <1> LD_FS_CDirLevel equ 127
3271                                    <1> LD_FS_CDIR_Converted equ 128
3272                                    <1>
3273                                    <1> ; Valid FAT Types
```

```
3274                            <1> FS_FAT12 equ 1
3275                            <1> FS_FAT16_CHS equ 2
3276                            <1> FS_FAT32_CHS equ 3
3277                            <1> FS_FAT16_LBA equ 4
3278                            <1> FS_FAT32_LBA equ 5
3279                            <1>
3280                            <1> ; Cursor Location
3281                            <1> CCCpointer equ  0450h   ; BIOS data, current cursor column
3282                            <1> ; FAT Clusters EOC sign
3283                            <1> FAT12EOC equ 0FFFh
3284                            <1> FAT16EOC equ 0FFFFh
3285                            <1> ;FAT32EOC equ 0FFFFFFFh ; It is not direct usable for 8086 code
3286                            <1> ; BAD Cluster
3287                            <1> FAT12BADC equ 0FF7h
3288                            <1> FAT16BADC equ 0FFF7h
3289                            <1> ;FAT32BADC equ 0FFFFFF7h ; It is not direct usable for 8086 code
3290                            <1> ; MS-DOS FAT16 FS (Maximum Possible) Last Cluster Number= 0FFF6h
3291                            <1>
3292                            <1> ; TRFS
3293                            <1>
3294                            <1> bs_FS_JmpBoot equ 0 ; jmp short bsBootCode
3295                            <1>                     ; db 0EBh, db 3Fh, db 90h
3296                            <1> bs_FS_Identifier equ 3  ; db 'FS', db 0
3297                            <1> bs_FS_BytesPerSec equ 6 ; dw 512
3298                            <1> bs_FS_MediaAttrib equ 8 ; db 3
3299                            <1> bs_FS_PartitionID equ 9 ; db 0A1h
3300                            <1> bs_FS_VersionMaj equ 10 ; db 01h
3301                            <1> bs_FS_VersionMin equ 11 ; db 0
3302                            <1> bs_FS_BeginSector equ 12   ; dd 0
3303                            <1> bs_FS_VolumeSize equ 16 ; dd 2880
3304                            <1> bs_FS_StartupFD equ 20 ; dd 0
3305                            <1> bs_FS_MATLocation equ 24 ; dd 1
3306                            <1> bs_FS_RootDirD equ 28 ; dd 8
3307                            <1> bs_FS_SystemConfFD equ 32 ; dd 0
3308                            <1> bs_FS_SwapFD equ 36 ; dd 0
3309                            <1> bs_FS_UnDelDirD equ 40 ; dd 0
3310                            <1> bs_FS_DriveNumber equ 44 ; db 0
3311                            <1> bs_FS_LBA_Ready equ 45 ; db 0
3312                            <1> bs_FS_MagicWord equ 46
3313                            <1> bs_FS_SecPerTrack equ 46 ; db 0A1h
3314                            <1> bs_FS_Heads equ 47 ; db 01h
3315                            <1> bs_FS_OperationSys equ 48 ; db "TR-SINGLIX v1.0b"
3316                            <1> bs_FS_Terminator equ 64 ; db 0
3317                            <1> bs_FS_BootCode equ 65
3318                            <1>
3319                            <1> FS_MAT_DATLocation equ 12
3320                            <1> FS_MAT_DATScount equ 16
3321                            <1> FS_MAT_FreeSectors equ 20
3322                            <1> FS_MAT_FirstFreeSector equ 24
3323                            <1> FS_RDT_VolumeSerialNo equ 28
3324                            <1> FS_RDT_VolumeName equ 64
3325                            <1>
3326                            <1> ; FAT12 + FAT16 + FAT32
3327                            <1> BS_JmpBoot equ 0
3328                            <1> BS_OEMName equ 3
3329                            <1> BPB_BytsPerSec equ 11
3330                            <1> BPB_SecPerClust equ 13
3331                            <1> BPB_RsvdSecCnt equ 14
3332                            <1> BPB_NumFATs equ 16
3333                            <1> BPB_RootEntCnt equ 17
3334                            <1> BPB_TotalSec16 equ 19
3335                            <1> BPB_Media equ 21
3336                            <1> BPB_FATSz16 equ 22
3337                            <1> BPB_SecPerTrk equ 24
3338                            <1> BPB_NumHeads equ 26
3339                            <1> BPB_HiddSec equ 28
3340                            <1> BPB_TotalSec32 equ 32
3341                            <1>
3342                            <1> ; FAT12 and FAT16 only
3343                            <1> BS_DrvNum equ 36
3344                            <1> BS_Reserved1 equ 37
3345                            <1> BS_BootSig equ 38
3346                            <1> BS_VolID equ 39
3347                            <1> BS_VolLab equ 43
3348                            <1> BS_FilSysType equ 54 ; 8 bytes
3349                            <1> BS_BootCode equ 62
3350                            <1>
3351                            <1> ; FAT32 only
3352                            <1> BPB_FATSz32 equ 36 ; FAT32, 4 bytes
3353                            <1> BPB_ExtFlags equ 40 ; FAT32, 2 bytes
3354                            <1> BPB_FSVer equ 42 ; FAT32, 2 bytes
3355                            <1> BPB_RootClus equ 44 ; FAT32, 4 bytes
3356                            <1> BPB_FSInfo equ 48 ; FAT 32, 2 bytes
3357                            <1> BPB_BkBootSec equ 50 ; FAT32, 2 bytes
3358                            <1> BPB_Reserved equ 52 ; FAT32, 12 bytes
3359                            <1> BS_FAT32_DrvNum equ 64 ; FAT32, 1 byte
3360                            <1> BS_FAT32_Reserved1 equ 65 ; FAT32, 1 byte
3361                            <1> BS_FAT32_BootSig equ 66 ; FAT32, 1 byte
3362                            <1> BS_FAT32_VolID equ 67 ; FAT32, 4 bytes
3363                            <1> BS_FAT32_VolLab equ 71 ; FAT32, 11 bytes
3364                            <1> BS_FAT32_FilSysType equ 82 ; FAT32, 8 bytes
3365                            <1> BS_FAT32_BootCode equ 90
3366                            <1>
3367                            <1> ; 29/02/2016
3368                            <1> ;(FAT32 Free Cluster Count & First Free Cluster values)
3369                            <1> ;[BPB_Reserved] = Free Cluster Count (offset 52)
3370                            <1> ;[BPB_Reserved+4] = First Free Cluster (offset 56)
3371                            <1>
3372                            <1> BS_Validation equ 510
3373                            <1>
3374                            <1> ; 15/02/2016
3375                            <1> ; FILE.ASM - 09/10/2011
3376                            <1> ; Directory Entry Structure
3377                            <1> ; 29/10/2009 (According to Microsoft FAT32 File System Specification)
3378                            <1> DirEntry_Name equ 0
```

```
                                <1> DirEntry_Attr equ 11
                                <1> DirEntry_NTRes equ 12
                                <1> DirEntry_CrtTimeTenth equ 13
                                <1> DirEntry_CrtTime equ 14
                                <1> DirEntry_CrtDate equ 16
                                <1> DirEntry_LastAccDate equ 18
                                <1> DirEntry_FstClusHI equ 20
                                <1> DirEntry_WrtTime equ 22
                                <1> DirEntry_WrtDate equ 24
                                <1> DirEntry_FstClusLO equ 26
                                <1> DirEntry_FileSize equ 28
                                    %include 'trdosk1.s' ; 04/01/2016
                                <1> ; ****************************************************************************
                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.4) - SYS INIT : trdosk1.s
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Last Update: 18/04/2021
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Beginning: 04/01/2016
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Assembler: NASM version 2.15 (trdos386.s)
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
                                <1> ; TRDOS2.ASM (09/11/2011)
                                <1> ; ****************************************************************************
                                <1> ; TRDOS2.ASM (c) 2004-2011 Erdogan TAN [ 17/01/2004 ] Last Update: 09/11/2011
                                <1> ;
                                <1>
                                <1> sys_init:
                                <1>       ; 18/04/2021 (TRDOS 386 v2.0.4)
                                <1>       ; 20/01/2018  (v2.0.1)
                                <1>       ; 23/01/2017  (v2.0.0)
                                <1>       ; 07/05/2016
                                <1>       ; 02/05/2016
                                <1>       ; 24/04/2016
                                <1>       ; 14/04/2016
                                <1>       ; 13/04/2016
                                <1>       ; 30/03/2016
                                <1>       ; 24/01/2016
                                <1>       ; 06/01/2016
                                <1>       ; 04/01/2016 (TRDOS 386 v2.0 - Beginning)
                                <1>
                                <1>       ; 23/01/2017 - reset timer frequency (to 18.2Hz)
00006F26 B036                   <1>       mov   al, 00110110b ; 36h
00006F28 E643                   <1>       out   43h, al
00006F2A 31C0                   <1>       xor   eax, eax  ; sub     al, al ; 0
00006F2C E640                   <1>       out   40h, al ; LB
00006F2E E640                   <1>       out   40h, al ; HB
                                <1>       ;
                                <1>       ; 30/03/2016
                                <1>       ; Clear Logical DOS Disk Description Tables Area
                                <1>       ;xor   eax, eax
00006F30 BF00010900             <1>       mov   edi, Logical_DOSDisks
00006F35 B980060000             <1>       mov   ecx, 6656/4 ; 26*256 = 6656 bytes
00006F3A F3AB                   <1>       rep   stosd ; 1664 times 4 bytes
                                <1>
00006F3C B83F3A2F00             <1>       mov   eax, '?:/'
00006F41 A3[6F810100]           <1>       mov   [Current_Dir_Drv], eax
                                <1>
                                <1>       ; Logical DRV INIT (only for hard disks)
00006F46 E809040000             <1>       call  ldrv_init  ; trdosk2.s
                                <1>
                                <1>       ; When floppy_drv_init call is disabled
                                <1>       ; media changed sign is needed
                                <1>       ; for proper drive initialization
                                <1>
00006F4B BE00010900             <1>       mov   esi, Logical_DOSDisks
00006F50 B001                   <1>       mov   al, 1 ; Initialization sign (invalid_fd_parameter)
00006F52 83C67E                 <1>       add   esi, LD_MediaChanged ; Media Change Status = 1 (init needed)
00006F55 8806                   <1>       mov   [esi], al ; A:
00006F57 81C600010000           <1>       add   esi, 100h
00006F5D 8806                   <1>       mov   [esi], al ; B:
                                <1>
                                <1> _current_drive_bootdisk:
00006F5F 8A15[EA680000]         <1>       mov   dl, [boot_drv] ; physical drive number
00006F65 80FAFF                 <1>       cmp   dl, 0FFh
00006F68 740A                   <1>       je    short _last_dos_diskno_check
                                <1> _boot_drive_check:
00006F6A 80FA80                 <1>       cmp   dl, 80h
00006F6D 7218                   <1>       jb    short _current_drive_a
00006F6F 80EA7E                 <1>       sub   dl, 7Eh ; C = 2 , D = 3
00006F72 EB13                   <1>       jmp   short _current_drive_a
                                <1>
                                <1> _last_dos_diskno_check:
00006F74 8A15[54390100]         <1>       mov   dl, [Last_DOS_DiskNo]
00006F7A 80FA02                 <1>       cmp   dl, 2
00006F7D 7706                   <1>       ja    short _current_drive_c
00006F7F 7406                   <1>       je    short _current_drive_a
00006F81 30D2                   <1>       xor   dl, dl ; A:
00006F83 EB02                   <1>       jmp   short _current_drive_a
                                <1>
                                <1> _current_drive_c:
00006F85 B202                   <1>       mov   dl, 2 ; C:
                                <1>
                                <1> _current_drive_a:
00006F87 8815[EB680000]         <1>       mov   [drv], dl
00006F8D BE[56390100]           <1>         mov     esi, msg_CRLF_temp
00006F92 E8AE000000             <1>       call  print_msg
                                <1>
00006F97 8A15[EB680000]         <1>       mov   dl, [drv]
                                <1> _default_drive_c:
00006F9D E8100C0000             <1>       call  change_current_drive
00006FA2 731C                   <1>       jnc   short _start_mainprog
                                <1>
                                <1> _drv_not_ready_error:
00006FA4 BE[113C0100]           <1>       mov   esi, msgl_drv_not_ready
```

```
3200 00006FA9 E897000000          <1>       call   print_msg
3201                              <1>         ;jmp _end_of_mainprog
3202                              <1>
3203                              <1>       ; 20/01/2018
3204 00006FAE B202                <1>       mov    dl, 2
3205 00006FB0 3815[EB680000]      <1>       cmp    [drv], dl
3206 00006FB6 736B                <1>       jnb    short _end_of_mainprog
3207 00006FB8 8815[EB680000]      <1>       mov    [drv], dl
3208 00006FBE EBDD                <1>       jmp    short _default_drive_c
3209                              <1>
3210                              <1> _start_mainprog:
3211                              <1>       ; 18/04/2021 (TRDOS 386 v2.0.4 - Beginning)
3212                              <1>       ; 07/01/2017
3213                              <1>       ; 07/05/2016
3214                              <1>       ; 02/05/2016
3215                              <1>       ; 24/04/2016 (TRDOS 386 v2)
3216                              <1>       ; Retro UNIX 386 v1, 'sys_init' (u0.s)
3217                              <1>       ; 23/06/2015
3218                              <1>
3219                              <1>       ; 02/05/2016
3220                              <1>       ; 24/04/2016
3221                              <1>       ;mov   ax, 1
3222                              <1>       ; 18/04/2021 - TRDOS 386 v2.0.4
3223 00006FC0 31C0                <1>       xor    eax, eax
3224 00006FC2 FEC0                <1>       inc    al  ; ax = 1
3225 00006FC4 A2[B3030300]        <1>       mov    [u.uno], al
3226 00006FC9 66A3[4E030300]      <1>       mov    [mpid], ax
3227 00006FCF 66A3[20000300]      <1>       mov    [p.pid], ax
3228 00006FD5 A2[B0000300]        <1>       mov    [p.stat], al
3229 00006FDA C605[A8030300]04    <1>       mov    byte [u.quant], time_count  ; 07/01/2017
3230                              <1>       ;
3231 00006FE1 A1[A8800100]        <1>       mov    eax, [k_page_dir]
3232 00006FE6 A3[B8030300]        <1>       mov    [u.pgdir], eax ; reset
3233                              <1>       ;
3234 00006FEB E885EAFFFF          <1>       call   allocate_page
3235 00006FF0 0F82B5000000        <1>       jc     panic
3236 00006FF6 A3[B4030300]        <1>       mov    [u.upage], eax ; user structure page
3237 00006FFB A3[C0000300]        <1>       mov    [p.upage], eax
3238 00007000 E8E1EAFFFF          <1>       call   clear_page
3239                              <1>       ;
3240                              <1>       ; 24/08/2015
3241 00007005 FE0D[5B030300]      <1>       dec    byte [sysflg] ; FFh = ready for system call
3242                              <1>                            ; 0 = executing a system call
3243                              <1>       ; 13/04/2016
3244                              <1>       ; Clear Environment Variables Page/Area
3245 0000700B BF00300900          <1>       mov    edi, Env_Page ; 93000h
3246 00007010 B980000000          <1>       mov    ecx, Env_Page_Size / 4    ; 512/4  (4096/4)

3247 00007015 31C0                <1>       xor    eax, eax
3248 00007017 F3AB                <1>       rep    stosd
3249                              <1>
3250                              <1>       ; 14/04/2016
3251 00007019 E8EB340000          <1>       call   mainprog_startup_configuration
3252                              <1>
3253 0000701E E8D00C0000          <1>         call    dos_prompt
3254                              <1>
3255                              <1> _end_of_mainprog:
3256 00007023 BE[56390100]        <1>         mov     esi, msg_CRLF_temp
3257 00007028 E818000000          <1>       call   print_msg
3258 0000702D BE[5C390100]        <1>       mov    esi, mainprog_Version
3259 00007032 E80E000000          <1>       call   print_msg
3260                              <1>       ; 24/01/2016
3261 00007037 28E4                <1>       sub    ah, ah
3262 00007039 E89D9EFFFF          <1>       call   int16h ; call getch
3263 0000703E E941A3FFFF          <1>       jmp    cpu_reset
3264                              <1>
3265 00007043 EBFE                <1> infinitiveloop: jmp short infinitiveloop
3266                              <1>
3267                              <1> print_msg:
3268                              <1>       ; 13/05/2016
3269                              <1>       ; 04/01/2016
3270                              <1>       ; 01/07/2015
3271                              <1>       ; 13/03/2015 (Retro UNIX 386 v1)
3272                              <1>       ; 07/03/2014 (Retro UNIX 8086 v1)
3273                              <1>       ; (Modified registers: EAX, EBX, ECX, EDX, ESI, EDI)
3274                              <1>       ;
3275 00007045 8A3D[D6800100]      <1>       mov    bh, [ACTIVE_PAGE] ; 04/01/2016 (ptty)
3276                              <1>       ;mov   bl, 07h ; Black background, light gray forecolor
3277                              <1>
3278 0000704B AC                  <1>       lodsb
3279                              <1> pmsg1:
3280 0000704C 56                  <1>       push   esi
3281                              <1>       ;mov   bh, [ACTIVE_PAGE] ; 04/01/2016 (ptty)
3282 0000704D B307                <1>       mov    bl, 07h ; Black background, light gray forecolor
3283 0000704F E80FB2FFFF          <1>       call   _write_tty
3284 00007054 5E                  <1>       pop    esi
3285 00007055 AC                  <1>       lodsb
3286 00007056 20C0                <1>       and    al, al
3287 00007058 75F2                <1>       jnz    short pmsg1
3288 0000705A C3                  <1>       retn
3289                              <1>
3290                              <1> clear_screen:
3291                              <1>       ; 06/12/2020
3292                              <1>       ; 03/12/2020 (TRDOS 386 v2.0.3)
3293                              <1>       ; 13/05/2016
3294                              <1>       ; 30/01/2016
3295                              <1>       ; 24/01/2016
3296                              <1>       ; 04/01/2016
3297 0000705B 0FB61D[D6800100]    <1>       movzx  ebx, byte [ACTIVE_PAGE] ; video page number (0 to 7)
3298 00007062 8AA3[CB6A0000]      <1>       mov    ah, [ebx+vmode] ; default = 03h (80x25 text)
3299 00007068 80FC04              <1>       cmp    ah, 4
3300 0000706B 7205                <1>       jb     short cls1
3301 0000706D 80FC07              <1>       cmp    ah, 7
3302 00007070 7530                <1>       jne    short vga_clear
3303                              <1> cls1:
```

```
3304                                    <1>       ;mov   bh, bl
3305                                    <1>       ;mov   bl, 7
3306 00007072 3A25[BA6A0000]            <1>       cmp    ah, [CRT_MODE] ; current video mode ?
3307 00007078 740E                      <1>       je     short cls2 ; yes (current video mode = 3)
3308                                    <1>       ;;call set_mode_3 ; set video mode to 3 (& clear screen)
3309                                    <1>       ;;retn
3310                                    <1>       ; 06/12/2020
3311 0000707A 803D[04120300]00          <1>       cmp    byte [pmi32], 0
3312 00007081 771F                      <1>       ja     short vga_clear
3313 00007083 E9E5B1FFFF                <1>       jmp    set_mode_3
3314                                    <1> cls2:
3315 00007088 88DF                      <1>       mov    bh, bl ; video page (0 to 7)
3316 0000708A B307                      <1>       mov    bl, 07h ; attribute to be used on blanked line
3317 0000708C 28C0                      <1>       sub    al, al ; 0 =  entire window
3318 0000708E 6631C9                    <1>       xor    cx, cx
3319 00007091 66BA4F18                  <1>       mov    dx, 184Fh
3320 00007095 E81AAFFFFF                <1>       call   _scroll_up ; 24/01/2016
3321                                    <1>       ;
3322                                    <1>       ;mov   bh, [ACTIVE_PAGE] ; video page number (0 to 7)
3323 0000709A 6631D2                    <1>       xor    dx, dx
3324                                    <1>       ;call  _set_cpos ; 24/01/2016
3325                                    <1>       ;;retn
3326                                    <1>       ; 03/12/2020
3327 0000709D E958B2FFFF                <1>       jmp    _set_cpos ; returns to the caller of this proc
3328                                    <1> ;cls3:
3329                                    <1> ;     retn
3330                                    <1>       ; 06/12/2020
3331                                    <1> vga_clear:
3332                                    <1>       ; 03/12/2020
3333                                    <1>       ; set mode by using _int10h
3334                                    <1>       ; (also clears screen)
3335 000070A2 88E0                      <1>       mov    al, ah
3336 000070A4 28E4                      <1>       sub    ah, ah  ; set current video mode
3337                                    <1>       ;call  _int10h ; simulates int 10h in TRDOS 386 kernel
3338                                    <1>       ;jmp   short cls3
3339 000070A6 E950A6FFFF                <1>       jmp    _int10h ; returns to the caller of this proc
3340                                    <1>
3341                                    <1> panic:
3342                                    <1>       ; 13/05/2016 (TRDOS 386 = TRDOS v2)
3343                                    <1>       ; 13/03/2015 (Retro UNIX 386 v1)
3344                                    <1>       ; 07/03/2014 (Retro UNIX 8086 v1)
3345 000070AB BE[FF430100]              <1>       mov    esi, panic_msg
3346 000070B0 E890FFFFFF                <1>       call   print_msg
3347                                    <1> key_to_reboot:
3348                                    <1>         ; 24/01/2016
3349 000070B5 28E4                      <1>         sub     ah, ah
3350 000070B7 E81F9EFFFF                <1>         call    int16h ; call   getch
3351                                    <1>         ; wait for a character from the current tty
3352                                    <1>       ;
3353 000070BC B00A                      <1>       mov    al, 0Ah
3354 000070BE 8A3D[D6800100]            <1>       mov    bh, [ptty] ; [ACTIVE_PAGE]
3355 000070C4 B307                      <1>       mov    bl, 07h ; Black background,
3356                                    <1>                    ; light gray forecolor
3357 000070C6 E898B1FFFF                <1>       call   _write_tty
3358 000070CB E9B4A2FFFF                <1>       jmp    cpu_reset
3359                                    <1>
3360                                    <1> ctrlbrk:
3361                                    <1>       ; 12/11/2015
3362                                    <1>       ; 13/03/2015 (Retro UNIX 386 v1)
3363                                    <1>       ; 06/12/2013 (Retro UNIX 8086 v1)
3364                                    <1>       ;
3365                                    <1>       ; INT 1Bh (control+break) handler
3366                                    <1>       ;
3367                                    <1>             ; Retro Unix 8086 v1 feature only!
3368                                    <1>             ;
3369 000070D0 66833D[AA030300]00        <1>       cmp    word [u.intr], 0
3370 000070D8 7645                      <1>       jna    short cbrk4
3371                                    <1> cbrk0:
3372                                    <1>       ; 12/11/2015
3373                                    <1>       ; 06/12/2013
3374 000070DA 66833D[AC030300]00        <1>       cmp    word [u.quit], 0
3375 000070E2 743B                      <1>       jz     short cbrk4
3376                                    <1>       ;
3377                                    <1>       ; 20/09/2013
3378 000070E4 6650                      <1>       push   ax
3379 000070E6 A0[D6800100]              <1>       mov    al, [ptty]
3380                                    <1>       ;
3381                                    <1>       ; 12/11/2015
3382                                    <1>       ;
3383                                    <1>       ; ctrl+break (EOT, CTRL+D) from serial port
3384                                    <1>       ; or ctrl+break from console (pseudo) tty
3385                                    <1>       ; (!redirection!)
3386                                    <1>       ;
3387 000070EB 3C08                      <1>       cmp    al, 8 ; serial port tty nums > 7
3388 000070ED 7211                      <1>         jb      short cbrk1 ; console (pseudo) tty
3389                                    <1>       ;
3390                                    <1>       ; Serial port interrupt handler sets [ptty]
3391                                    <1>       ; to the port's tty number (as temporary).
3392                                    <1>       ;
3393                                    <1>       ; If active process is using a stdin or
3394                                    <1>       ; stdout redirection (by the shell),
3395                                    <1>         ; console tty keyboard must be available
3396                                    <1>       ; to terminate running process,
3397                                    <1>       ; in order to prevent a deadlock.
3398                                    <1>       ;
3399 000070EF 52                        <1>       push   edx
3400 000070F0 0FB615[B3030300]          <1>       movzx  edx, byte [u.uno]
3401 000070F7 3A82[7F000300]            <1>       cmp     al, [edx+p.ttyc-1] ; console tty (rw)
3402 000070FD 5A                        <1>       pop    edx
3403 000070FE 7412                      <1>       je     short cbrk2
3404                                    <1> cbrk1:
3405 00007100 FEC0                      <1>       inc    al  ; [u.ttyp] : 1 based tty number
3406                                    <1>       ; 06/12/2013
3407 00007102 3A05[94030300]            <1>       cmp    al, [u.ttyp] ; recent open tty (r)
3408 00007108 7408                      <1>       je     short cbrk2
```

```
3409 0000710A 3A05[95030300]      <1>         cmp     al, [u.ttyp+1] ; recent open tty (w)
3410 00007110 750B                <1>       jne   short cbrk3
3411                              <1> cbrk2:
3412                              <1>       ;; 06/12/2013
3413                              <1>       ;mov   ax, [u.quit]
3414                              <1>       ;and   ax, ax
3415                              <1>       ;jz    short cbrk3
3416                              <1>       ;
3417 00007112 6631C0              <1>       xor    ax, ax ; 0
3418 00007115 6648                <1>       dec    ax
3419                              <1>       ; 0FFFFh = 'ctrl+brk' keystroke
3420 00007117 66A3[AC030300]      <1>       mov    [u.quit], ax
3421                              <1> cbrk3:
3422 0000711D 6658                <1>       pop    ax
3423                              <1> cbrk4:
3424 0000711F C3                  <1>       retn
3425                              <1>
3426                              <1>
3427                              <1> ; 31/12/2017
3428                              <1> ; TRDOS 386 - 30/12/2017
3429                              <1> %define get_rtc_date RTC_40
3430                              <1> %define get_rtc_time RTC_20
3431                              <1> %define     set_rtc_date RTC_50
3432                              <1> %define set_rtc_time RTC_30
3433                              <1> get_rtc_date_time:
3434                              <1> ; Retro UNIX 8086 v1 - UNIX.ASM (01/09/2014)
3435                              <1> ;epoch:
3436                              <1>       ; 18/04/2021 (TRDOS 386 v2.0.3)
3437                              <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
3438                              <1>       ; 15/03/2015 (Retro UNIX 386 v1 - 32 bit version)
3439                              <1>       ; 09/04/2013 (Retro UNIX 8086 v1 - UNIX.ASM)
3440                              <1>       ; 'epoch' procedure prototype:
3441                              <1>       ;                   UNIXCOPY.ASM, 10/03/2013
3442                              <1>       ; 14/11/2012
3443                              <1>       ; unixboot.asm (boot file configuration)
3444                              <1>       ; version of "epoch" procedure in "unixproc.asm"
3445                              <1>       ; 21/7/2012
3446                              <1>       ; 15/7/2012
3447                              <1>       ; 14/7/2012
3448                              <1>       ; Erdogan Tan - RETRO UNIX v0.1
3449                              <1>       ; compute current date and time as UNIX Epoch/Time
3450                              <1>       ; UNIX Epoch: seconds since 1/1/1970 00:00:00
3451                              <1>       ;
3452                              <1>         ;  ((Modified registers: EAX, EDX, ECX, EBX))
3453                              <1>       ;
3454                              <1>
3455                              <1>       ; 18/04/2021
3456                              <1>       ; INPUT:
3457                              <1>       ;     none (real time clock)
3458                              <1>       ; OUTPUT:
3459                              <1>       ;     eax = unix epoch time value
3460                              <1>       ;         (seconds since 1/1/1970 00:00:00)
3461                              <1>
3462 00007120 E882F4FFFF          <1>       call   get_rtc_time       ; Return Current Time
3463                              <1>         ;xchg      ch, cl ; 18/04/2021
3464 00007125 66890D[767D0100]    <1>         mov  [hour], cx ; BCD, cl = minute, ch = hour
3465                              <1>         ;xchg      dh, dl ; 18/04/2021
3466                              <1>       ;mov   [second], dx ; BCD, dh = second, dl = dse
3467                              <1>       ; 18/04/2021
3468 0000712C 8835[7A7D0100]      <1>       mov    [second], dh ; second
3469                              <1>       ;
3470 00007132 E8E1F4FFFF          <1>         call       get_rtc_date       ; Return Current Date
3471                              <1>         ;xchg      ch, cl ; 18/04/2021
3472 00007137 66890D[707D0100]    <1>         mov  [year], cx ; BCD, cl = year, ch = century
3473                              <1>         ;xchg      dh, dl ; 18/04/2021
3474 0000713E 668915[727D0100]    <1>         mov  [month], dx ; BCD, dl = day, dh = month
3475                              <1>       ;
3476                              <1>       ;mov   al, [hour] ; Hour
3477                              <1>         ; 18/04/2021
3478 00007145 A0[777D0100]        <1>       mov    al, [hour+1] ; Hour
3479                              <1>              ; AL <-- BCD number
3480 0000714A D410                <1>         db   0D4h, 10h          ; Undocumented inst. AAM
3481                              <1>                                  ; AH = AL / 10h
3482                              <1>                                  ; AL = AL MOD 10h
3483 0000714C D50A                <1>         aad ; AX= AH*10+AL
3484                              <1>       ;mov   [hour], al
3485                              <1>       ;mov   al, [hour+1] ; Minute
3486 0000714E 8605[767D0100]      <1>       xchg   al, [hour] ; [hour] = hour, al = minute
3487                              <1>              ; AL <-- BCD number
3488 00007154 D410                <1>         db   0D4h, 10h          ; Undocumented inst. AAM
3489                              <1>                                  ; AH = AL / 10h
3490                              <1>                                  ; AL = AL MOD 10h
3491 00007156 D50A                <1>         aad ; AX= AH*10+AL
3492 00007158 A2[787D0100]        <1>       mov    [minute], al
3493 0000715D A0[7A7D0100]        <1>       mov    al, [second] ; Second
3494                              <1>              ; AL <-- BCD number
3495 00007162 D410                <1>         db   0D4h, 10h          ; Undocumented inst. AAM
3496                              <1>                                  ; AH = AL / 10h
3497                              <1>                                  ; AL = AL MOD 10h
3498 00007164 D50A                <1>         aad ; AX= AH*10+AL
3499 00007166 A2[7A7D0100]        <1>       mov    [second], al
3500 0000716B 66A1[707D0100]      <1>       mov    ax, [year] ; Year (century)
3501                              <1>       ; 18/04/2021
3502                              <1>       ;push  eax ; puhs ax
3503                              <1>       ;mov   al, ah ; century ; 18/04/2021
3504                              <1>              ; AL <-- BCD number
3505 00007171 D410                <1>         db   0D4h, 10h          ; Undocumented inst. AAM
3506                              <1>                                  ; AH = AL / 10h
3507                              <1>                                  ; AL = AL MOD 10h
3508 00007173 D50A                <1>         aad ; AX= AH*10+AL
3509                              <1>       ;mov   ah, 100
3510                              <1>       ;mul   ah
3511                              <1>       ;mov   [year], ax
3512                              <1>       ; 18/04/2021
3513                              <1>       ; ax = al = year (0 to 99)
```

```
3514 00007175 668705[707D0100]    <1>       xchg   ax, [year] ; [year+1] = century -> ah
3515                              <1>       ;pop   eax ; pop ax
3516 0000717C 88E0                <1>       mov    al, ah  ; century
3517                              <1>              ; AL <-- BCD number
3518 0000717E D410                <1>         db   0D4h, 10h           ; Undocumented inst. AAM
3519                              <1>                                  ; AH = AL / 10h
3520                              <1>                                  ; AL = AL MOD 10h
3521 00007180 D50A                <1>         aad ; AX= AH*10+AL
3522                              <1>       ; 18/04/2021
3523 00007182 B464                <1>       mov    ah, 100 ; 100*(century byte of year)
3524 00007184 F6E4                <1>       mul    ah
3525                              <1>       ;
3526 00007186 660105[707D0100]    <1>       add    [year], ax
3527                              <1>       ;mov   al, [month] ; Month
3528                              <1>       ; 18/04/2021
3529 0000718D A0[737D0100]        <1>       mov    al, [month+1] ; Month
3530                              <1>              ; AL <-- BCD number
3531 00007192 D410                <1>         db   0D4h, 10h           ; Undocumented inst. AAM
3532                              <1>                                  ; AH = AL / 10h
3533                              <1>                                  ; AL = AL MOD 10h
3534 00007194 D50A                <1>         aad ; AX= AH*10+AL
3535                              <1>       ;mov   [month], al
3536                              <1>         ;mov     al, [month+1]         ; Day
3537                              <1>       ; 18/04/2021
3538 00007196 8605[727D0100]      <1>       xchg   al, [month] ; [month] = month, al = day
3539                              <1>              ; AL <-- BCD number
3540 0000719C D410                <1>         db   0D4h, 10h           ; Undocumented inst. AAM
3541                              <1>                                  ; AH = AL / 10h
3542                              <1>                                  ; AL = AL MOD 10h
3543 0000719E D50A                <1>         aad ; AX= AH*10+AL
3544 000071A0 A2[747D0100]        <1>         mov     [day], al
3545                              <1>
3546 000071A5 C3                  <1>       retn   ; 30/12/2017
3547                              <1>
3548                              <1> epoch:
3549 000071A6 E875FFFFFF          <1>       call   get_rtc_date_time ; TRDOS 386 - 30/12/2017
3550                              <1>
3551                              <1> convert_to_epoch:
3552                              <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
3553                              <1>       ; 15/03/2015 (Retro UNIX 386 v1 - 32 bit modification)
3554                              <1>       ; 09/04/2013 (Retro UNIX 8086 v1)
3555                              <1>       ;
3556                              <1>       ; ((Modified registers: EAX, EDX, EBX))
3557                              <1>       ;
3558                              <1>       ; Derived from DALLAS Semiconductor
3559                              <1>       ; Application Note 31 (DS1602/DS1603)
3560                              <1>       ; 6 May 1998
3561 000071AB 29C0                <1>       sub    eax, eax
3562 000071AD 66A1[707D0100]      <1>       mov    ax, [year]
3563 000071B3 662DB207            <1>       sub    ax, 1970
3564 000071B7 BA6D010000          <1>       mov    edx, 365
3565 000071BC F7E2                <1>       mul    edx
3566 000071BE 31DB                <1>       xor    ebx, ebx
3567 000071C0 8A1D[727D0100]      <1>       mov    bl, [month]
3568 000071C6 FECB                <1>       dec    bl
3569 000071C8 D0E3                <1>       shl    bl, 1
3570                              <1>       ;sub   edx, edx
3571 000071CA 668B93[7C7D0100]    <1>       mov    dx, [EBX+DMonth]
3572 000071D1 8A1D[747D0100]      <1>         mov     bl, [day]
3573 000071D7 FECB                <1>       dec    bl
3574 000071D9 01D0                <1>       add    eax, edx
3575 000071DB 01D8                <1>       add    eax, ebx
3576                              <1>                     ; EAX = days since 1/1/1970
3577 000071DD 668B15[707D0100]    <1>       mov    dx, [year]
3578 000071E4 6681EAB107          <1>       sub    dx, 1969
3579 000071E9 66D1EA              <1>       shr    dx, 1
3580 000071EC 66D1EA              <1>       shr    dx, 1
3581                              <1>              ; (year-1969)/4
3582 000071EF 01D0                <1>       add    eax, edx
3583                              <1>                     ; + leap days since 1/1/1970
3584 000071F1 803D[727D0100]02    <1>       cmp    byte [month], 2     ; if past february
3585 000071F8 7610                <1>       jna    short cte1
3586 000071FA 668B15[707D0100]    <1>       mov    dx, [year]
3587 00007201 6683E203            <1>       and    dx, 3 ; year mod 4
3588 00007205 7503                <1>       jnz    short cte1
3589                              <1>                     ; and if leap year
3590 00007207 83C001              <1>       add    eax, 1       ; add this year's leap day (february 29)
3591                              <1> cte1:                      ; compute seconds since 1/1/1970
3592 0000720A BA18000000          <1>       mov    edx, 24
3593 0000720F F7E2                <1>       mul    edx
3594 00007211 8A15[767D0100]      <1>       mov    dl, [hour]
3595 00007217 01D0                <1>       add    eax, edx
3596                              <1>              ; EAX = hours since 1/1/1970 00:00:00
3597                              <1>       ;mov   ebx, 60
3598 00007219 B33C                <1>       mov    bl, 60
3599 0000721B F7E3                <1>       mul    ebx
3600 0000721D 8A15[787D0100]      <1>       mov    dl, [minute]
3601 00007223 01D0                <1>       add    eax, edx
3602                              <1>              ; EAX = minutes since 1/1/1970 00:00:00
3603                              <1>       ;mov   ebx, 60
3604 00007225 F7E3                <1>       mul    ebx
3605 00007227 8A15[7A7D0100]      <1>       mov    dl, [second]
3606 0000722D 01D0                <1>       add    eax, edx
3607                              <1>              ; EAX -> seconds since 1/1/1970 00:00:00
3608 0000722F C3                  <1>       retn
3609                              <1>
3610                              <1> ;set_date_time:
3611                              <1> convert_from_epoch:
3612                              <1>       ; 18/04/2021 (v2.0.4)
3613                              <1>       ; 31/12/2017 (v2.0.0)
3614                              <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
3615                              <1>       ; 15/03/2015 (Retro UNIX 386 v1 - 32 bit version)
3616                              <1>       ; 20/06/2013 (Retro UNIX 8086 v1)
3617                              <1>       ; 'convert_from_epoch' procedure prototype:
3618                              <1>       ;                 UNIXCOPY.ASM, 10/03/2013
```

```
3619                                    <1>     ;
3620                                    <1>     ; ((Modified registers: EAX, EDX, ECX, EBX))
3621                                    <1>     ;
3622                                    <1>     ; Derived from DALLAS Semiconductor
3623                                    <1>     ; Application Note 31 (DS1602/DS1603)
3624                                    <1>     ; 6 May 1998
3625                                    <1>     ;
3626                                    <1>     ; INPUT:
3627                                    <1>     ; EAX = Unix (Epoch) Time
3628                                    <1>     ;
3629 00007230 31D2                     <1>     xor   edx, edx
3630 00007232 B93C000000               <1>     mov   ecx, 60
3631 00007237 F7F1                     <1>     div   ecx
3632                                    <1>     ;mov  [imin], eax   ; whole minutes
3633                                    <1>                         ; since 1/1/1970
3634 00007239 668915[7A7D0100]         <1>     mov   [second], dx  ; leftover seconds
3635 00007240 29D2                     <1>     sub   edx, edx
3636 00007242 F7F1                     <1>     div   ecx
3637                                    <1>     ;mov  [ihrs], eax   ; whole hours
3638                                    <1>     ;                   ; since 1/1/1970
3639 00007244 668915[787D0100]         <1>     mov   [minute], dx  ; leftover minutes
3640 0000724B 31D2                     <1>     xor   edx, edx
3641                                    <1>     ;mov  cx, 24
3642 0000724D B118                     <1>     mov   cl, 24
3643 0000724F F7F1                     <1>     div   ecx
3644                                    <1>     ;mov  [iday], ax    ; whole days
3645                                    <1>                         ; since 1/1/1970
3646 00007251 668915[767D0100]         <1>     mov   [hour], dx    ; leftover hours
3647 00007258 05DB020000               <1>     add   eax, 365+366 ; whole day since
3648                                    <1>                         ; 1/1/1968
3649                                    <1>     ;mov  [iday], ax
3650 0000725D 50                       <1>     push  eax
3651 0000725E 29D2                     <1>     sub   edx, edx
3652 00007260 B9B5050000               <1>     mov   ecx, (4*365)+1 ; 4 years = 1461 days
3653 00007265 F7F1                     <1>     div   ecx
3654 00007267 59                       <1>     pop   ecx
3655                                    <1>     ;mov  [lday], ax    ; count of quadyrs (4 years)
3656                                    <1>     ;push dx
3657                                    <1>     ; 18/04/2021
3658 00007268 52                       <1>     push  edx
3659                                    <1>     ;mov  [qday], dx    ; days since quadyr began
3660 00007269 6683FA3C                 <1>     cmp   dx, 31 + 29   ; if past feb 29 then
3661 0000726D F5                       <1>     cmc                 ; add this quadyr's leap day
3662 0000726E 83D000                   <1>     adc   eax, 0        ; to # of qadyrs (leap days)
3663                                    <1>     ;mov  [lday], ax    ; since 1968
3664                                    <1>     ;mov  cx, [iday]
3665 00007271 91                       <1>     xchg  ecx, eax      ; ECX = lday, EAX = iday
3666 00007272 29C8                     <1>     sub   eax, ecx      ; iday - lday
3667 00007274 B96D010000               <1>     mov   ecx, 365
3668 00007279 31D2                     <1>     xor   edx, edx
3669                                    <1>     ; EAX = iday-lday, EDX = 0
3670 0000727B F7F1                     <1>     div   ecx
3671                                    <1>     ;mov  [iyrs], ax    ; whole years since 1968
3672                                    <1>     ;jday = iday - (iyrs*365) - lday
3673                                    <1>     ;mov  [jday], dx    ; days since 1/1 of current year
3674                                    <1>     ;add  eax, 1968
3675 0000727D 6605B007                 <1>     add   ax, 1968      ; compute year
3676 00007281 66A3[707D0100]           <1>     mov   [year], ax
3677 00007287 6689D1                   <1>     mov   cx, dx
3678                                    <1>     ;;mov dx, [qday]
3679                                    <1>     ;pop  dx
3680                                    <1>     ; 18/04/2021
3681 0000728A 5A                       <1>     pop   edx
3682 0000728B 6681FA6D01               <1>     cmp   dx, 365           ; if qday <= 365 and qday >= 60
3683 00007290 7709                     <1>     ja    short cfe1    ; jday = jday +1
3684 00007292 6683FA3C                 <1>     cmp   dx, 60        ; if past 2/29 and leap year then
3685 00007296 F5                       <1>       cmc               ; add a leap day to the # of whole
3686 00007297 6683D100                 <1>     adc   cx, 0         ; days since 1/1 of current year
3687                                    <1> cfe1:
3688                                    <1>     ;mov  [jday], cx
3689                                    <1>     ;mov  bx, 12        ; estimate month
3690                                    <1>     ; 18/04/2021
3691 0000729B 29DB                     <1>     sub   ebx, ebx
3692 0000729D B30C                     <1>     mov   bl, 12
3693 0000729F 66BA6E01                 <1>     mov   dx, 366       ; mday, max. days since 1/1 is 365
3694 000072A3 6683E003                 <1>     and   ax, 11b       ; year mod 4 (and dx, 3)
3695                                    <1> cfe2: ; Month calculation  ; 0 to 11  (11 to 0)
3696 000072A7 6639D1                   <1>     cmp   cx, dx        ; mday = # of days passed from 1/1
3697 000072AA 731C                     <1>     jnb   short cfe3
3698                                    <1>     ;dec  bx            ; month = month - 1
3699                                    <1>     ;shl  bx, 1
3700                                    <1>     ; 18/04/2021
3701 000072AC FECB                     <1>     dec   bl
3702 000072AE D0E3                     <1>     shl   bl, 1
3703 000072B0 668B93[7C7D0100]         <1>     mov   dx, [EBX+DMonth] ; # elapsed days at 1st of month
3704                                    <1>     ; 18/04/2021
3705                                    <1>     ;shr  bx, 1         ; bx = month - 1 (0 to 11)
3706 000072B7 D0EB                     <1>     shr   bl, 1
3707                                    <1>     ;cmp  bx, 1         ; if month > 2 and year mod 4  = 0
3708 000072B9 80FB01                   <1>     cmp   bl, 1
3709 000072BC 76E9                     <1>     jna   short cfe2    ; then mday = mday + 1
3710 000072BE 76E7                     <1>     jna   short cfe2    ; then mday = mday + 1
3711 000072C0 08C0                     <1>     or    al, al        ; if past 2/29 and leap year then
3712 000072C2 75E3                     <1>     jnz   short cfe2    ; add leap day (to mday)
3713 000072C4 6642                     <1>     inc   dx            ; mday = mday + 1
3714 000072C6 EBDF                     <1>     jmp   short cfe2
3715                                    <1> cfe3:
3716                                    <1>     ;inc  bx          ; -> bx = month, 1 to 12
3717                                    <1>     ; 18/04/2021
3718 000072C8 FEC3                     <1>     inc   bl
3719 000072CA 66891D[727D0100]         <1>     mov   [month], bx
3720 000072D1 6629D1                   <1>     sub   cx, dx      ; day = jday - mday + 1
3721                                    <1>     ;inc  cx
3722                                    <1>     ; 18/04/2021
3723 000072D4 FEC1                     <1>     inc   cl
```

```
3724                                    <1>       ;mov   [day], cx
3725 000072D6 880D[747D0100]            <1>       mov    [day], cl
3726                                    <1>
3727                                    <1>       ; eax, ebx, ecx, edx is changed at return
3728                                    <1>       ; output ->
3729                                    <1>       ; [year], [month], [day], [hour], [minute], [second]
3730                                    <1>
3731 000072DC C3                       <1>       retn   ; 31/12/2017 (TRDOS 386)
3732                                    <1>
3733                                    <1> set_rtc_date_time:
3734                                    <1>       ; 31/12/2017 (v2.0.0)
3735                                    <1>       ; 30/12/2017 (TRDOS 386)
3736                                    <1>       ; 15/03/2015 (Retro UNIX 386 v1 - 32 bit version)
3737                                    <1>       ; 20/06/2013 (Retro UNIX 8086 v1)
3738 000072DD E80F000000               <1>       call   set_date_bcd
3739                                    <1>       ; Set real-time clock date
3740 000072E2 E85EF3FFFF               <1>       call   set_rtc_date ; RTC_50
3741                                    <1>       ; Set real-time clock time
3742 000072E7 E832000000               <1>       call   set_time_bcd
3743 000072EC E9E5F2FFFF               <1>       jmp    set_rtc_time ; RTC_30
3744                                    <1>
3745                                    <1> ; 31/12/2017
3746                                    <1> set_date_bcd:
3747 000072F1 A0[717D0100]              <1>         mov     al, [year+1]
3748 000072F6 D40A                     <1>       aam    ; ah = al / 10, al = al mod 10
3749 000072F8 D510                     <1>       db     0D5h, 10h    ; Undocumented inst. AAD
3750                                    <1>                           ; AL = AH * 10h + AL
3751 000072FA 88C5                     <1>       mov    ch, al ; century (BCD)
3752 000072FC A0[707D0100]             <1>       mov    al, [year]
3753 00007301 D40A                     <1>       aam    ; ah = al / 10, al = al mod 10
3754 00007303 D510                     <1>       db     0D5h, 10h    ; Undocumented inst. AAD
3755                                    <1>                           ; AL = AH * 10h + AL
3756 00007305 88C1                     <1>       mov    cl, al ; year (BCD)
3757 00007307 A0[727D0100]             <1>         mov  al, [month]
3758 0000730C D40A                     <1>       aam    ; ah = al / 10, al = al mod 10
3759 0000730E D510                     <1>       db     0D5h, 10h    ; Undocumented inst. AAD
3760                                    <1>                           ; AL = AH * 10h + AL
3761 00007310 88C6                     <1>       mov    dh, al ; month (BCD)
3762 00007312 A0[747D0100]             <1>       mov    al, [day]
3763 00007317 D40A                     <1>       aam    ; ah = al / 10, al = al mod 10
3764 00007319 D510                     <1>       db     0D5h, 10h    ; Undocumented inst. AAD
3765                                    <1>                           ; AL = AH * 10h + AL
3766                                    <1>       ; 18/04/2021
3767 0000731B 88C2                     <1>       mov    dl, al ; day (BCD)
3768 0000731D C3                       <1>       retn   ; 30/12/2017
3769                                    <1>
3770                                    <1> ; 31/12/2017
3771                                    <1> set_time_bcd:
3772                                    <1>         ; Read real-time clock time
3773                                    <1>       ; (get day light saving time bit status)
3774 0000731E FA                       <1>       cli
3775 0000731F E850F4FFFF               <1>       CALL   UPD_IPR             ; CHECK FOR UPDATE IN PROCESS
3776                                    <1>       ; cf = 1 -> al = 0
3777 00007324 7207                     <1>         jc      short stime1
3778 00007326 B00B                     <1>       MOV    AL,CMOS_REG_B       ; ADDRESS ALARM REGISTER
3779 00007328 E87DF4FFFF               <1>       CALL   CMOS_READ           ; READ CURRENT VALUE OF DSE BIT
3780                                    <1> stime1:
3781 0000732D FB                       <1>       sti
3782 0000732E 2401                     <1>       AND    AL,00000001B        ; MASK FOR VALID DSE BIT
3783 00007330 88C2                     <1>       MOV    DL,AL               ; SET [DL] TO ZERO FOR NO DSE BIT
3784                                    <1>       ; DL = 1 or 0 (day light saving time)
3785                                    <1>       ;
3786 00007332 A0[767D0100]             <1>       mov    al, [hour]
3787 00007337 D40A                     <1>       aam    ; ah = al / 10, al = al mod 10
3788 00007339 D510                     <1>       db     0D5h,10h     ; Undocumented inst. AAD
3789                                    <1>                           ; AL = AH * 10h + AL
3790 0000733B 88C5                     <1>       mov    ch, al ; hour (BCD)
3791 0000733D A0[787D0100]             <1>         mov     al, [minute]
3792 00007342 D40A                     <1>       aam    ; ah = al / 10, al = al mod 10
3793 00007344 D510                     <1>       db     0D5h,10h     ; Undocumented inst. AAD
3794                                    <1>                           ; AL = AH * 10h + AL
3795 00007346 88C1                     <1>       mov    cl, al       ; minute (BCD)
3796 00007348 A0[7A7D0100]             <1>         mov     al, [second]
3797 0000734D D40A                     <1>       aam    ; ah = al / 10, al = al mod 10
3798 0000734F D510                     <1>       db     0D5h,10h     ; Undocumented inst. AAD
3799                                    <1>                           ; AL = AH * 10h + AL
3800 00007351 88C6                     <1>       mov    dh, al      ; second (BCD)
3801 00007353 C3                       <1>       retn   ; 30/12/2017
3107                                        %include 'trdosk2.s' ; 04/01/2016
3108                                    <1> ; ****************************************************************************
3109                                    <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.2) - DRV INIT : trdosk2.s
3110                                    <1> ; ----------------------------------------------------------------------------
3111                                    <1> ; Last Update: 30/08/2020
3112                                    <1> ; ----------------------------------------------------------------------------
3113                                    <1> ; Beginning: 04/01/2016
3114                                    <1> ; ----------------------------------------------------------------------------
3115                                    <1> ; Assembler: NASM version 2.14 (trdos386.s)
3116                                    <1> ; ----------------------------------------------------------------------------
3117                                    <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
3118                                    <1> ; TRDOS2.ASM (09/11/2011)
3119                                    <1> ; ****************************************************************************
3120                                    <1> ; DRV_INIT.ASM (c) 2009-2011 Erdogan TAN  [26/09/2009] Last Update: 07/08/2011
3121                                    <1> ;
3122                                    <1>
3123                                    <1> ldrv_init: ; Logical Drive Initialization
3124                                    <1>       ; 30/08/2020
3125                                    <1>       ; 25/08/2020
3126                                    <1>       ; 11/08/2020, 13/08/2020
3127                                    <1>       ; 17/07/2020, 20/07/2020
3128                                    <1>       ; 14/07/2020, 15/07/2020
3129                                    <1>       ; 30/01/2018
3130                                    <1>       ; 27/12/2017
3131                                    <1>       ; 12/02/2016
3132                                    <1>       ; 06/01/2016
3133                                    <1>       ;      ('diskinit.inc', 'diskio.inc' integration)
```

```
3134                                    <1>       ; 04/01/2016 (TRDOS 386 = TRDOS v2.0)
3135                                    <1>       ; 07/08/2011
3136                                    <1>       ; 20/09/2009
3137                                    <1>       ; 2005
3138                                    <1>
3139                                    <1>       ; 15/07/2020
3140                                    <1>       ;movzx ecx, byte [HF_NUM] ; number of fixed disks
3141                                    <1>       ;cmp   cl, 1
3142                                    <1>       ;jnb   short load_hd_partition_tables
3143                                    <1>
3144 00007354 A0[44810100]             <1>       mov    al, [HF_NUM] ; number of fixed disks
3145 00007359 20C0                     <1>       and    al, al
3146 0000735B 7501                     <1>       jnz    short load_hd_partition_tables
3147                                    <1>
3148                                    <1>       ; no any hard disks
3149 0000735D C3                       <1>       retn
3150                                    <1>
3151                                    <1> load_hd_partition_tables:
3152                                    <1>       ;mov   esi, [HDPM_TBL_VEC] ; primary master disk FDPT
3153                                    <1>       ; 15/07/2020
3154 0000735E BE[48810100]             <1>       mov    esi, HDPM_TBL_VEC
3155 00007363 BF[6E850100]             <1>       mov    edi, PTable_hd0
3156 00007368 B280                     <1>       mov    dl, 80h
3157                                    <1>       ; 15/07/2020
3158 0000736A A2[ED680000]             <1>       mov    [hdc], al
3159                                    <1>       ;xor   ecx, ecx ; 0
3160                                    <1> load_next_hd_partition_table:
3161                                    <1>       ; 20/07/2020
3162 0000736F 31C9                     <1>       xor    ecx, ecx ; 0
3163                                    <1>       ;push  ecx
3164 00007371 57                       <1>       push   edi ; *
3165                                    <1>       ;push  esi ; FDPT (+ DPTE) address
3166                                    <1>       ; 15/07/2020
3167 00007372 AD                       <1>       lodsd
3168 00007373 56                       <1>       push   esi ; ** ; next FDPT (+ DPTE) address ptr
3169                                    <1>
3170                                    <1>       ;mov   al, [esi+20] ; DPTE offset 4
3171                                    <1>       ;and   al, 40h ;  LBA bit (bit 6)
3172                                    <1>       ;;shr  al, 6
3173                                    <1>       ;mov   [HD_LBA_yes], al
3174                                    <1>
3175                                    <1>       ; 15/07/2020
3176 00007374 8A4814                   <1>       mov    cl, [eax+20]
3177 00007377 80E140                   <1>       and    cl, 40h
3178 0000737A 880D[72860100]           <1>       mov    [HD_LBA_yes], cl
3179                                    <1>
3180 00007380 E844040000               <1>       call   load_masterboot
3181                                    <1>       ;jc    short pass_pt_this_hard_disk
3182                                    <1>       ; 13/08/2020
3183 00007385 0F828A000000             <1>       jc     pass_pt_this_hard_disk
3184                                    <1>
3185 0000738B BB[2C850100]             <1>       mov    ebx, PartitionTable
3186 00007390 89DE                     <1>       mov    esi, ebx
3187                                    <1>       ;mov   ecx, 16
3188 00007392 B110                     <1>       mov    cl, 16
3189 00007394 F3A5                     <1>       rep    movsd
3190 00007396 89DE                     <1>       mov    esi, ebx
3191                                    <1>       ;mov   byte [hdc], 4 ; 4 - partition index
3192                                    <1>       ; 15/07/2020
3193 00007398 C605[73860100]04         <1>       mov    byte [PP_Counter], 4
3194                                    <1> loc_validate_hdp_partition:
3195                                    <1>       ;cmp   byte [esi+ptFileSystemID], 0
3196                                    <1>       ;jna   short loc_validate_next_hdp_partition2
3197                                    <1>       ; 13/08/2020
3198 0000739F 8A4604                   <1>       mov    al, [esi+ptFileSystemID]
3199 000073A2 20C0                     <1>       and    al, al
3200 000073A4 7457                     <1>       jz     short loc_validate_next_hdp_partition2
3201                                    <1>
3202 000073A6 56                       <1>       push   esi ; *** ; Masterboot partition table offset
3203 000073A7 52                       <1>       push   edx ; **** ; dl = Physical drive number
3204                                    <1>
3205                                    <1>       ; 13/08/2020
3206 000073A8 3C05                     <1>       cmp    al, 05h  ; Extended partition CHS
3207 000073AA 7404                     <1>       je     short loc_set_ep_counter
3208 000073AC 3C0F                     <1>       cmp    al, 0Fh  ; Extended partition LBA
3209 000073AE 7511                     <1>       jne    short loc_validate_next_hdp_partition0
3210                                    <1>
3211                                    <1>       ;;inc  byte [PP_Counter]
3212                                    <1>       ; 15/07/2020
3213                                    <1>       ;inc   byte [EP_Counter] ; disk has valid partition(s)
3214                                    <1>
3215                                    <1> loc_set_ep_counter:
3216                                    <1>       ; 13/08/2020
3217 000073B0 803D[74860100]80         <1>       cmp    byte [EP_Counter], 80h
3218 000073B7 7342                     <1>       jnb    short loc_validate_next_hdp_partition1
3219                                    <1>
3220 000073B9 8815[74860100]           <1>       mov    byte [EP_Counter], dl ; disk drv has extd. part.
3221                                    <1>
3222 000073BF EB3A                     <1>       jmp    short loc_validate_next_hdp_partition1
3223                                    <1>
3224                                    <1> loc_validate_next_hdp_partition0:
3225 000073C1 31FF                     <1>       xor    edi, edi ; 0
3226                                    <1>       ; Input -> ESI = PartitionTable offset
3227                                    <1>       ; DL = Hard disk drive number
3228                                    <1>       ; EDI = 0 -> Primary Partition
3229                                    <1>       ; EDI > 0 -> Extended Partition's Start Sector
3230 000073C3 E885010000               <1>       call   validate_hd_fat_partition
3231 000073C8 730E                     <1>       jnc    short loc_set_valid_hdp_partition_entry
3232                                    <1>
3233                                    <1>       ;pop   edx
3234                                    <1>       ;push  edx
3235 000073CA 8B1424                   <1>       mov    edx, [esp]  ; ****
3236 000073CD 8B742404                 <1>       mov    esi, [esp+4] ; *** ; 30/01/2018
3237 000073D1 E8D1020000               <1>       call   validate_hd_fs_partition
3238 000073D6 7223                     <1>       jc     short loc_validate_next_hdp_partition1
```

```
                                <1> loc_set_valid_hdp_partition_entry:
000073D8 8A0D[54390100]         <1>       mov    cl, [Last_DOS_DiskNo]
000073DE 80C141                 <1>       add    cl, 'A'
                                <1>       ; ESI = Logical dos drive description table address
000073E1 880E                   <1>       mov    [esi+LD_Name], cl
                                <1>       ; 15/07/2020
000073E3 8A4602                 <1>       mov    al, [esi+LD_PhyDrvNo] ; Physical drive number
                                <1>       ;mov   al, [esp] ; ****
000073E6 2C7F                   <1>       sub    al, 7Fh
                                <1>              ; AL = 1 to 4
000073E8 C0E002                 <1>       shl    al, 2 ; AL = 4 to 16
                                <1>
000073EB 8A15[73860100]         <1>       mov    dl, [PP_Counter]
                                <1>
                                <1>       ;sub   al, [PP_Counter]
000073F1 28D0                   <1>       sub    al, dl ; [PP_Counter] ; 4 - partition index
                                <1>
                                <1>       ; AL = Partition entry/index, 0 based
                                <1>       ;  0 -> hd 0, Partition Table offset = 0
                                <1>       ; 15 -> hd 3, Partition Table offset = 3
                                <1>
                                <1>       ;mov   [esi+LD_PartitionEntry], al
                                <1>
                                <1>       ; 15/07/2020
000073F3 B404                   <1>       mov    ah, 4
                                <1>       ;sub   ah, [PP_Counter]
000073F5 28D4                   <1>       sub    ah, dl
                                <1>
                                <1>       ; AH = Primary partition index, 0 to 3 ; pt entry
                                <1>       ;            (4 to 7 for logical disk partitions)
                                <1>
                                <1>       ;mov   [esi+LD_DParamEntry], ah
000073F7 6689467C               <1>       mov    [esi+LD_PartitionEntry], ax
                                <1>
                                <1> loc_validate_next_hdp_partition1:
000073FB 5A                     <1>       pop    edx ; **** ; dl = Physical drive number
000073FC 5E                     <1>       pop    esi ; *** ; Masterboot partition table offset
                                <1>
                                <1> loc_validate_next_hdp_partition2:
                                <1>       ; ESI = PartitionTable offset
                                <1>       ; DL = Hard/Fixed disk drive number
                                <1>
                                <1>       ;dec   byte [hdc] ; 4 - partition index
                                <1>       ;jz    short pass_pt_this_hard_disk
                                <1>       ; 15/07/2020
000073FD FE0D[73860100]         <1>       dec    byte [PP_Counter] ; 4 - partition index
00007403 7410                   <1>       jz     short pass_pt_this_hard_disk
                                <1>
00007405 83C610                 <1>       add    esi, 16 ; 10h
00007408 EB95                   <1>       jmp    short loc_validate_hdp_partition
                                <1>
                                <1> loc_not_any_extd_partitions:
                                <1>       ; 15/07/2020
0000740A C3                     <1>       retn
                                <1>
                                <1> loc_next_hd_partition_table:
0000740B FEC2                   <1>       inc    dl
                                <1>       ; 15/07/2020
                                <1>       ;add   esi, 32 ; next FDPT address
0000740D 83C740                 <1>       add    edi, 64 ; next partition table destination
00007410 E95AFFFFFF             <1>         jmp     load_next_hd_partition_table
                                <1>
                                <1> pass_pt_this_hard_disk:
                                <1>       ;pop   esi ; FDPT (+ DPTE) address
                                <1>       ; 15/07/2020
00007415 5E                     <1>       pop    esi ; ** ; next FDPT (+ DPTE) address ptr
00007416 5F                     <1>       pop    edi ; * ; Ptable_hd?
                                <1>       ;pop   ecx
                                <1>       ;loop  loc_next_hd_partition_table
00007417 FE0D[ED680000]         <1>       dec    byte [hdc]
0000741D 75EC                   <1>       jnz    short loc_next_hd_partition_table
                                <1>
                                <1>       ;cmp   byte [PP_Counter], 1
                                <1>       ;jnb   short load_extended_dos_partitions
                                <1>       ;; Empty partition table
                                <1>       ;retn
                                <1>
                                <1>       ; 11/08/2020
                                <1>       ; 17/07/2020
                                <1> check_extended_partitions:
                                <1>       ; 15/07/2020
                                <1>       ;cmp   byte [EP_Counter], 0
                                <1>       ;jna   short loc_not_any_extd_partitions
                                <1>       ; 13/08/2020
0000741F A0[74860100]           <1>       mov    al, [EP_Counter] ; 1st disk drv has extd partition
00007424 08C0                   <1>       or     al, al ; 0 ?
00007426 74E2                   <1>       jz     short loc_not_any_extd_partitions
                                <1>
                                <1> load_extended_dos_partitions:
                                <1>       ;mov   byte [hdc], 80h
                                <1>       ; 13/08/2020
00007428 A2[ED680000]           <1>       mov    byte [hdc], al ; 1st disk drv has extd partition
                                <1>       ; 25/08/2020
0000742D 2C80                   <1>       sub    al, 80h
0000742F 740E                   <1>       jz     short loc_set_ext_ptable_hd0
00007431 C0E006                 <1>       shl    al, 6 ; * 64
00007434 0FB6F0                 <1>       movzx  esi, al
00007437 81C6[6E850100]         <1>       add    esi, PTable_hd0
0000743D EB05                   <1>       jmp    short next_hd_extd_partition
                                <1>
                                <1>       ; 25/08/2020
                                <1> loc_set_ext_ptable_hd0:
0000743F BE[6E850100]           <1>       mov    esi, PTable_hd0
                                <1>
                                <1> next_hd_extd_partition:
```

```
3344                                 <1>       ; 17/07/2020
3345                                 <1>       ;mov  byte [EP_Counter], 0 ; Reset for each physical disk
3346                                 <1>       ; 13/08/2020
3347                                 <1>       ;mov  byte [LD_Counter], 0 ; Reset logical drive index
3348 00007444 66C705[74860100]00-   <1>       mov   word [EP_Counter], 0 ; Reset EP index and LD index
3348 0000744C 00                     <1>
3349                                 <1>
3350 0000744D 56                     <1>       push  esi ; **** ; PTable_hd? offset
3351                                 <1>
3352 0000744E C605[73860100]04       <1>       mov   byte [PP_Counter], 4
3353                                 <1>                           ; set for each extd partition table
3354                                 <1>       ;;mov  ecx, 4
3355                                 <1>       ;mov  cl, 4
3356 00007455 8A15[ED680000]         <1>       mov   dl, [hdc]
3357                                 <1> hd_check_fs_id_05h:
3358 0000745B 8A4604                 <1>       mov   al, [esi+ptFileSystemID]
3359 0000745E 3C05                   <1>       cmp   al, 05h ; Is it an extended dos partition ?
3360 00007460 7411                   <1>       je    short loc_set_ep_start_sector ; yes
3361                                 <1> hd_check_fs_id_0Fh:
3362 00007462 3C0F                   <1>       cmp   al, 0Fh ; Is it an extended win4 (LBA mode) partition ?
3363 00007464 740D                   <1>       je    short loc_set_ep_start_sector ; yes
3364                                 <1>
3365                                 <1> continue_to_check_ep:
3366                                 <1>       ;add  esi, 16
3367                                 <1>       ;loop hd_check_fs_id_05h
3368                                 <1>       ; 15/07/2020
3369                                 <1>       ;dec  cl
3370                                 <1>       ;jz   short continue_check_ep_next_disk
3371 00007466 FE0D[73860100]         <1>       dec   byte [PP_Counter] ; 4 --> 0
3372 0000746C 742D                   <1>       jz    short continue_check_ep_next_disk
3373 0000746E 83C610                 <1>       add   esi, 16
3374 00007471 EBE8                   <1>       jmp   short hd_check_fs_id_05h
3375                                 <1>
3376                                 <1> loc_set_ep_start_sector:
3377                                 <1>       ; dl = [hdc] ; Drive number
3378                                 <1>       ; 15/07/2020
3379 00007473 8B4E08                 <1>       mov   ecx, [esi+ptStartSector]
3380                                 <1>       ; 30/08/2020
3381 00007476 890D[76860100]         <1>       mov   [MBR_EP_StartSector], ecx
3382                                 <1>       ; 20/07/2020
3383                                 <1> loc_validate_hde_partition_next:
3384 0000747C 890D[7A860100]         <1>       mov   [EP_StartSector], ecx ; Extended partition's start sector
3385 00007482 BB[6E830100]           <1>         mov ebx, MasterBootBuff
3386 00007487 803D[72860100]01       <1>       cmp   byte [HD_LBA_yes], 1 ; LBA ready = Yes
3387 0000748E 7227                   <1>       jb    short loc_hd_load_ep_05h ; cf = 1 ; 20/07/2020
3388                                 <1>       ; 11/08/2020
3389                                 <1>       ; (BugFix for extended partition type 05h beyond CHS limit)
3390                                 <1>       ; (Infact if extended partition starts at the beyond of CHS limit,
3391                                 <1>       ;  it's partition ID must be 0Fh but they/somebodies had used 05h.)
3392                                 <1>       ;cmp  al, 05h
3393                                 <1>       ;je   short loc_hd_load_ep_05h
3394                                 <1> loc_hd_load_ep_0Fh:
3395                                 <1>       ; 04/01/2016
3396                                 <1>       ;push ecx
3397                                 <1>       ; 15/07/2020
3398                                 <1>       ;mov  ecx, [esi+ptStartSector] ; sector number
3399                                 <1>       ;mov  ebx, MasterBootBuff ; buffer address
3400                                 <1>       ; LBA read/write (with private LBA function)
3401                                 <1>       ;((Retro UNIX 386 v1 - DISK I/O code by Erdogan Tan))
3402                                 <1>       ; dl = physical drive number (0,1, 80h, 81h, 82h, 83h)
3403                                 <1>       ;mov  ah, 1Bh ; LBA read
3404                                 <1>       ;mov  al, 1 ; sector count
3405 00007490 66B8011B               <1>       mov   ax, 1B01h
3406 00007494 E8CADDFFFF             <1>       call  int13h
3407                                 <1>       ;pop  ecx
3408                                 <1>       ;jnc  short loc_hd_move_ep_table
3409                                 <1>       ; 15/07/2020
3410 00007499 732E                   <1>       jnc   short loc_validate_hde_partition
3411                                 <1>
3412                                 <1> continue_check_ep_next_disk:
3413                                 <1>       ; 15/07/2020
3414                                 <1>       ;pop  edi ; PTable_ep?
3415 0000749B 5E                     <1>       pop   esi ; **** ; PTable_hd?
3416 0000749C A0[44810100]           <1>       mov   al, [HF_NUM] ; number of hard disks
3417 000074A1 047F                   <1>       add   al, 7Fh
3418 000074A3 3805[ED680000]         <1>       cmp   [hdc], al
3419 000074A9 730B                   <1>       jnb   short loc_validating_hd_partitions_ok
3420 000074AB 83C640                 <1>       add   esi, 64
3421                                 <1>       ; 15/07/2020
3422                                 <1>       ;add  edi, 64
3423 000074AE FE05[ED680000]         <1>       inc   byte [hdc]
3424 000074B4 EB8E                   <1>       jmp   short next_hd_extd_partition
3425                                 <1>
3426                                 <1> loc_validating_hd_partitions_ok:
3427                                 <1>       ; 15/07/2020
3428                                 <1>       ;mov  al, [Last_DOS_DiskNo]
3429                                 <1> loc_drv_init_retn:
3430 000074B6 C3                     <1>       retn
3431                                 <1>
3432                                 <1> loc_hd_load_ep_05h:
3433                                 <1>       ; 20/07/2020 ('diskio.s', int13h, cf = 1 -> bugfix)
3434                                 <1>       ;clc    ; (Bug: int13h would not clear carry flag bit,
3435                                 <1>       ;       ; even if there would not be an error)
3436                                 <1>       ;       ; ((Fix: now, int13h procedure clears carry flag
3437                                 <1>       ;       ;  at the entrance of it.. 20/07/2020))
3438                                 <1>       ; 15/07/2020
3439                                 <1>       ;push ecx
3440 000074B7 8A7601                 <1>       mov   dh, [esi+ptBeginHead]
3441 000074BA 668B4E02               <1>         mov     cx, [esi+ptBeginSector]
3442 000074BE 66B80102               <1>       mov   ax, 0201h ; Read 1 sector
3443                                 <1>       ;mov  ebx, MasterBootBuff
3444 000074C2 E89CDDFFFF             <1>       call  int13h ; 20/07/2020
3445                                 <1>                     ; 'diskio.s' modification, 'clc'
3446                                 <1>       ;pop  ecx
3447 000074C7 72D2                   <1>       jc    short continue_check_ep_next_disk
```

```
                            <1>       ; 15/07/2020
                            <1>       ;jmp   short loc_validate_hde_partition
                            <1>
                            <1>       ; 15/07/2020
                            <1> ;loc_hd_move_ep_table:
                            <1>       ;;pop  edi
                            <1>       ;;push edi  ; PTable_ep?
                            <1>       ;mov   edi, [esp]
                            <1>         ;mov esi, PartitionTable ; Extended
                            <1>       ;mov   ebx, esi
                            <1>       ;;mov  ecx, 16
                            <1>       ;mov   cl, 16
                            <1>              ;rep   movsd
                            <1>       ;mov   esi, ebx
                            <1> ;loc_set_hde_sub_partition_count:
                            <1>       ;mov   byte [PP_Counter], 4
                            <1>       ;mov   byte [EP_Counter], 0
                            <1>
                            <1> loc_validate_hde_partition:
                            <1>       ; 13/08/2020
                            <1>       ; 15/07/2020
                            <1>       ;mov   byte [PP_Counter], 4
000074C9 BE[2C850100]       <1>       mov    esi, PartitionTable ; (in MasterBootBuff)
                            <1>       ; 13/08/2020
                            <1>       ;jmp   short get_minidisk_partition_entry
                            <1>
                            <1> ;get_minidisk_partition_entry:
                            <1> ;     ; 20/07/2020
                            <1> ;     cmp    byte [esi+ptFileSystemID], 0
                            <1> ;     ja     short loc_validate_minidisk_partition
                            <1> ;     ; 13/08/2020
                            <1> ;     jmp    short continue_check_ep_next_disk
                            <1>
                            <1> ;     ; 11/08/2020
                            <1> ;get_minidisk_partition_entry_next:
                            <1> ;     ; 13/08/2020
                            <1> ;     ;dec   byte [PP_Counter]
                            <1> ;     ;jz    short continue_check_ep_next_disk
                            <1> ;     ; 20/07/2020
                            <1> ;;get_minidisk_partition_entry_next:
                            <1> ;     ; 13/08/2020
                            <1> ;     cmp    esi, PartitionTable+64
                            <1> ;     jnb    short continue_check_ep_next_disk
                            <1> ;
                            <1> ;     add    esi, 16 ; 10h
                            <1> ;     ;jmp   short get_minidisk_partition_entry
                            <1>
                            <1>       ; 13/08/2020
                            <1> get_minidisk_partition_entry:
                            <1>       ; 20/07/2020
000074CE 807E0400           <1>       cmp    byte [esi+ptFileSystemID], 0
000074D2 76C7               <1>       jna    short continue_check_ep_next_disk ; 13/08/2020
                            <1>
                            <1> loc_validate_minidisk_partition:
                            <1>       ; 13/08/2020
                            <1>       ; 20/07/2020
                            <1>       ;push  esi ; *** ; Extended partition table offset
                            <1>
                            <1>       ; 13/08/2020
000074D4 FE05[74860100]     <1>       inc    byte [EP_Counter] ; current (sub partition) index
                            <1>                              ; in current extended partition
                            <1>
000074DA BF[7A860100]       <1>       mov    edi, EP_StartSector
                            <1>
                            <1>       ; Input -> ESI = PartitionTable offset
                            <1>       ; DL = Hard disk drive number
                            <1>       ; EDI = Extended partition start sector pointer
000074DF E869000000         <1>       call   validate_hd_fat_partition
                            <1>       ;pop   ecx ; *
000074E4 7308               <1>       jnc    short loc_set_valid_hde_partition_entry
                            <1>                     ; jump down to deep !!!
                            <1>
                            <1>       ;pop   esi ; *** ; Extended partition table offset
                            <1>       ; 13/08/2020
                            <1>       ;mov   esi, PartitionTable
                            <1>
                            <1>       ; 11/08/2020
                            <1>       ; ESI = Extended partition table offset
000074E6 8A15[ED680000]     <1>       mov    dl, [hdc]
                            <1>
                            <1>       ;; DL = Hard disk drive number
                            <1>       ;dec   byte [PP_Counter]
                            <1>       ;jz    short continue_check_ep_next_disk
                            <1>       ;add   esi, 16 ; 10h
                            <1>       ;mov   dl, [hdc]
                            <1>       ;jmp   short get_minidisk_partition_entry
                            <1>
                            <1>       ; 11/08/2020
                            <1>       ;jmp   short get_minidisk_partition_entry_next
                            <1>
                            <1>       ; 23/08/2020
000074EC EB3D               <1>       jmp    short validate_next_minidisk_partition_ok
                            <1>
                            <1>       ; 17/07/2020
                            <1>       ;; jumping down to deep levels !!!
                            <1>       ; ((That is a pitty microsoft preferred ep table chain
                            <1>       ; instead of a single table as mbr partition table!?))
                            <1>
                            <1> loc_set_valid_hde_partition_entry:
                            <1>       ; 15/07/2020
000074EE A0[ED680000]       <1>       mov    al, [hdc] ; Hard disk drive number (>=80h)
000074F3 88C2               <1>       mov    dl, al ; mov dl, [hdc]
000074F5 2C7F               <1>       sub    al, 7Fh
                            <1>                  ; 1 to 4
000074F7 C0E002             <1>       shl    al, 2 ; 4 to 16
```

```
3553 000074FA 2A05[73860100]      <1>       sub    al, [PP_Counter] ; al - (4 - partition index)
3554                              <1>                 ; (disk number * 4) + partition index
3555                              <1>
3556                              <1>       ; AL = Partition entry/index, 0 based
3557                              <1>         ;  0 -> hd 0, Partition Table offset = 0
3558                              <1>         ; 15 -> hd 3, Partition Table offset = 3
3559                              <1>
3560                              <1>       ;mov   ah, 4 ; Logical dos partition (>= 4)
3561                              <1>       ;add   ah, [EP_Counter]
3562                              <1>              ; Logical disk partition index = 4 to 7
3563                              <1>              ; (Primary disk partition index = 0 to 3)
3564                              <1>
3565                              <1>       ; 13/08/2020
3566 00007500 8A25[75860100]      <1>       mov    ah, [LD_Counter] ; Logical drive index number
3567                              <1>                               ; (in current extended partition)
3568 00007506 80C404              <1>       add    ah, 4 ; 4 to 7
3569                              <1>
3570                              <1>       ; 15/07/2020
3571                              <1>       ; CX -> AX
3572                              <1>       ;; 06/01/2016 (TRDOS v2.0)
3573                              <1>       ;; BUGFIX *
3574                              <1>       ;;mov  [esi+LD_PartitionEntry], cl
3575                              <1>       ;;mov  [esi+LD_DParamEntry], ch
3576                              <1>       ;mov   [esi+LD_PartitionEntry], cx
3577 00007509 6689467C            <1>       mov    [esi+LD_PartitionEntry], ax
3578                              <1>
3579 0000750D 8A0D[54390100]      <1>       mov    cl, [Last_DOS_DiskNo]
3580 00007513 80C141              <1>       add    cl, 'A'
3581 00007516 880E                <1>       mov    [esi+LD_Name], cl
3582                              <1>
3583                              <1>       ; 17/07/2020
3584                              <1>       ;cmp   cl, 'Z'
3585                              <1>       ;jb    short logical_drive_count_ok_for_next
3586                              <1>       ;pop   esi ; ***
3587                              <1>       ;pop   esi ; ****
3588                              <1>       ;retn
3589                              <1>
3590                              <1> ;logical_drive_count_ok_for_next:
3591                              <1>
3592                              <1>       ;; 15/07/2020
3593                              <1>       ;inc   byte [EP_Counter]
3594                              <1>       ; 13/08/2020
3595 00007518 FE05[75860100]      <1>       inc    byte [LD_Counter]
3596                              <1>
3597                              <1>       ;mov   dl, [hdc]
3598                              <1>
3599                              <1>       ; 17/07/2020
3600                              <1>       ;; Now,
3601                              <1>       ;; we are swimming in deep of an extended partition !!!
3602                              <1>       ; (! sub or chained extended partition tables !)
3603                              <1>       ; ((Logical dos partitions in extended partition were called
3604                              <1>       ;  as 'mini disk partition' in msdos 6.0 source code.))
3605                              <1>
3606                              <1> validate_next_minidisk_partition:
3607                              <1>       ; 13/08/2020
3608                              <1>       ;pop   esi ; *** ; Extended partition table offset
3609                              <1>
3610                              <1>       ; 17/07/2020
3611                              <1>       ;cmp   byte [EP_Counter], 4
3612                              <1>       ; 13/08/2020
3613 0000751E 803D[75860100]04    <1>       cmp    byte [LD_Counter], 4 ; maximum 4 logical disks
3614                              <1>                                   ; per extended partition
3615 00007525 0F8370FFFFFF        <1>       jnb    continue_check_ep_next_disk
3616                              <1>
3617                              <1> validate_next_minidisk_partition_ok:
3618                              <1>       ; 13/08/2020
3619                              <1>       ;dec   byte [PP_Counter] ; 4 --> 0
3620                              <1>       ;jz    continue_check_ep_next_disk
3621                              <1>
3622                              <1>       ;cmp   esi, PartitionTable+64
3623                              <1>       ;jnb   continue_check_ep_next_disk
3624                              <1>
3625                              <1>       ;add   esi, 16
3626                              <1>       ; 13/08/2020
3627 0000752B BE[3C850100]        <1>       mov    esi, PartitionTable+16
3628                              <1>
3629                              <1>       ; 20/07/2020
3630 00007530 8A4604              <1>       mov    al, [esi+ptFileSystemID]
3631                              <1>
3632                              <1>       ; 20/07/2020
3633 00007533 3C05                <1>       cmp    al, 05h ; Is it an extended dos partition ?
3634 00007535 7408                <1>       je     short loc_minidisk_next_ep_lba_chs ; 17/07/2020
3635 00007537 3C0F                <1>       cmp    al, 0Fh ; Is it an extended win4 (LBA mode) partition ?
3636 00007539 0F855CFFFFFF        <1>       jne    continue_check_ep_next_disk ; AL must be 0 here
3637                              <1>                                        ; (when it is not 05h or 0Fh)
3638                              <1>                                        ; If AL is not ZERO -> EP Bug!
3639                              <1>                                        ; (!Microsoft DOS convention!)
3640                              <1> loc_minidisk_next_ep_lba_chs:
3641                              <1>       ; 17/07/2020
3642 0000753F 8B4E08              <1>       mov    ecx, [esi+ptStartSector] ; relative start sector number
3643                              <1>       ;add   ecx, [EP_StartSector]
3644                              <1>       ; 30/08/2020
3645 00007542 030D[76860100]      <1>       add    ecx, [MBR_EP_StartSector]
3646                              <1>       ; 20/07/2020
3647 00007548 E92FFFFFFF          <1>       jmp    loc_validate_hde_partition_next
3648                              <1>
3649                              <1> validate_hd_fat_partition:
3650                              <1>       ; 17/07/2020
3651                              <1>       ; 15/07/2020
3652                              <1>       ;      (optimization)
3653                              <1>       ; 14/07/2020
3654                              <1>       ;      (fat16 -big- partition search bugfix)
3655                              <1>       ; 27/12/2017
3656                              <1>       ; 12/02/2016
3657                              <1>       ; 07/01/2016 (TRDOS 386 = TRDOS v2.0)
```

```
                        <1>       ; 07/08/2011
                        <1>       ; 23/07/2011
                        <1>       ; Input
                        <1>       ;   DL = Hard/Fixed Disk Drive Number
                        <1>       ;   ESI = PartitionTable offset
                        <1>       ;   EDI = Extend. Part. Start Sector Pointer
                        <1>       ;   EDI = 0 -> Primary Partition
                        <1>       ;   byte [Last_DOS_DiskNo]
                        <1>       ; Output
                        <1>       ;  cf=0 -> Validated
                        <1>       ;   ESI = Logical dos drv desc. table
                        <1>       ;   EBX = FAT boot sector buffer
                        <1>       ;   byte [Last_DOS_DiskNo]
                        <1>       ;  cf=1 -> Not a valid FAT partition
                        <1>       ; EAX, EDX, ECX, EDI -> changed
                        <1>
                        <1>       ;mov   esi, PartitionTable
0000754D 8A6604         <1>       mov    ah, [esi+ptFileSystemID]
00007550 B002           <1>       mov    al, 2 ; 27/12/2017
00007552 80FC06         <1>       cmp    ah, 06h ; FAT16 CHS partition (>=32MB)
                        <1>       ; 12/02/2016
                        <1>       ;;jb   short loc_not_a_valid_fat_partition2
                        <1>       ;jnb   short vhdp_FAT16_32
                        <1>       ; 14/07/2020 (BugFix)
00007555 7711           <1>       ja     short vhdp_FAT16_32
00007557 7425           <1>       je     short loc_set_valid_hd_partition_params
                        <1>
                        <1> vhdp_FAT12_16:
                        <1>       ; 27/12/2017
00007559 FEC8           <1>       dec    al ; mov al, 1
0000755B 38C4           <1>       cmp    ah, al ; 1 ; FAT12 partition
0000755D 741F           <1>       je     short loc_set_valid_hd_partition_params
                        <1>       ;
0000755F FEC0           <1>       inc    al ; mov al, 2
00007561 80FC04         <1>       cmp    ah, 04h ; FAT16 CHS partition (< 32MB)
00007564 7418           <1>       je     short loc_set_valid_hd_partition_params
                        <1>
                        <1>       ; 15/07/2020
                        <1>       ; (ah = 05h, 02h or 03h)
                        <1> loc_not_a_valid_fat_partition1:
00007566 F9             <1>       stc
                        <1>       ; cf=1
00007567 C3             <1>       retn
                        <1>
                        <1> vhdp_FAT16_32:
                        <1>       ; 15/07/2020
                        <1>       ;mov   al, 3
00007568 FEC0           <1>       inc    al
0000756A 80FC0C         <1>       cmp    ah, 0Ch ; FAT32 LBA partition
0000756D 740F           <1>       je     short loc_set_valid_hd_partition_params
0000756F 7706           <1>       ja     short vhdp_check_FAT16_lba
                        <1>
                        <1> vhdp_check_FAT32_chs:
00007571 80FC0B         <1>       cmp    ah, 0Bh ; FAT32 CHS partition
00007574 7408           <1>       je     short loc_set_valid_hd_partition_params
                        <1>       ;jne   short loc_not_a_valid_fat_partition1
                        <1>
                        <1>       ;stc
                        <1> loc_not_a_valid_fat_partition2:
00007576 C3             <1>       retn
                        <1>
                        <1> vhdp_check_FAT16_lba:
00007577 80FC0E         <1>       cmp    ah, 0Eh ; FAT16 LBA partition
0000757A 75EA           <1>       jne    short loc_not_a_valid_fat_partition1
                        <1>
                        <1>       ;mov   al, 2
0000757C FEC8           <1>       dec    al
                        <1>
                        <1> loc_set_valid_hd_partition_params:
                        <1>       ; 15/07/2020
                        <1>       ;inc   byte [Last_DOS_DiskNo] ; > 1
                        <1>       ;
0000757E 31DB           <1>       xor    ebx, ebx
00007580 8A3D[54390100] <1>       mov    bh, [Last_DOS_DiskNo] ; * 256
00007586 FEC7           <1>       inc    bh ; 15/07/2020
00007588 81C300010900   <1>       add    ebx, Logical_DOSDisks
                        <1>       ;
0000758E C6430102       <1>       mov    byte [ebx+LD_DiskType], 2
00007592 885302         <1>       mov    byte [ebx+LD_PhyDrvNo], dl
                        <1>       ;mov   byte [ebx+LD_FATType], al ; 2 or 3
                        <1>       ;mov   byte [ebx+LD_FSType], ah ; 06h, 0Eh, 0Bh, 0Ch
00007595 66894303       <1>       mov    word [ebx+LD_FATType], ax
                        <1>       ;
00007599 8B4E08         <1>       mov    ecx, [esi+ptStartSector]
0000759C 09FF           <1>       or     edi, edi
0000759E 7402           <1>       jz     short pass_hd_FAT_ep_start_sector_adding
                        <1> loc_add_hd_FAT_ep_start_sector:
                        <1>       ; 17/07/2020
000075A0 030F           <1>       add    ecx, [edi]
                        <1> pass_hd_FAT_ep_start_sector_adding:
000075A2 894B6C         <1>       mov    [ebx+LD_StartSector], ecx
                        <1> loc_hd_FAT_logical_drv_init:
000075A5 89DD           <1>       mov    ebp, ebx
                        <1>       ;mov   dl, [ebx+LD_PhyDrvNo]
000075A7 A0[72860100]   <1>       mov    al, [HD_LBA_yes] ; 07/01/2016
000075AC 884305         <1>       mov    [ebx+LD_LBAYes], al
000075AF BB[7E860100]   <1>       mov    ebx, DOSBootSectorBuff ; buffer address
000075B4 08C0           <1>       or     al, al
000075B6 740C           <1>       jz     short loc_hd_FAT_drv_init_load_bs_chs
                        <1> loc_hd_FAT_drv_init_load_bs_lba:
                        <1>       ; DL = Physical drive number
                        <1>       ;mov   ecx, [esi+ptStartSector] ; sector number
                        <1>       ;mov   ebx, DOSBootSectorBuff ; buffer address
                        <1>       ; LBA read/write (with private LBA function)
                        <1>       ;((Retro UNIX 386 v1 - DISK I/O code by Erdogan Tan))
```

```
                                <1>       ; dl = physical drive number (0,1, 80h, 81h, 82h, 83h)
000075B8 B41B                   <1>       mov   ah, 1Bh ; LBA read
000075BA B001                   <1>       mov   al, 1 ; sector count
000075BC E8A2DCFFFF             <1>       call  int13h
000075C1 7313                   <1>       jnc   short loc_hd_drv_FAT_boot_validation
                                <1> loc_not_a_valid_fat_partition3:
000075C3 C3                     <1>       retn
                                <1> loc_hd_FAT_drv_init_load_bs_chs:
000075C4 8A7601                 <1>       mov   dh, [esi+ptBeginHead]
000075C7 668B4E02               <1>       mov   cx, [esi+ptBeginSector]
000075CB 66B80102               <1>       mov   ax, 0201h ; Read 1 sector
                                <1>       ;mov  ebx, DOSBootSectorBuff
000075CF E88FDCFFFF             <1>       call  int13h
000075D4 72ED                   <1>       jc    short loc_not_a_valid_fat_partition3
                                <1> loc_hd_drv_FAT_boot_validation:
                                <1>       ;mov  esi, DOSBootSectorBuff
000075D6 89DE                   <1>       mov   esi, ebx
000075D8 6681BEFE01000055AA     <1>       cmp   word [esi+BS_Validation], 0AA55h
000075E1 7512                   <1>       jne   short loc_not_a_valid_fat_partition4
000075E3 807E15F8               <1>       cmp   byte [esi+BPB_Media], 0F8h
000075E7 750C                   <1>       jne   short loc_not_a_valid_fat_partition4
                                <1>
                                <1>       ; 27/12/2017
000075E9 807D0303               <1>       cmp   byte [ebp+LD_FATType], 3
000075ED 7508                   <1>       jne   short loc_hd_FAT16_BPB
                                <1>
                                <1> loc_hd_drv_FAT32_boot_validation:
000075EF 807E4229               <1>       cmp   byte [esi+BS_FAT32_BootSig], 29h
000075F3 7416                   <1>       je    short loc_hd_FAT32_BPB
                                <1>
                                <1> loc_not_a_valid_fat_partition4:
000075F5 F9                     <1>       stc
000075F6 C3                     <1>       retn
                                <1>
                                <1> loc_hd_FAT16_BPB:
000075F7 807E2629               <1>       cmp   byte [esi+BS_BootSig], 29h
000075FB 75F8                   <1>       jne   short loc_not_a_valid_fat_partition4
                                <1>
000075FD 66837E1600             <1>       cmp   word [esi+BPB_FATSz16], 0
00007602 7607                   <1>       jna   short loc_hd_big_FAT16_BPB
00007604 B920000000             <1>       mov   ecx, 32
00007609 EB05                   <1>       jmp   short loc_hd_move_FAT_BPB
                                <1>
                                <1> loc_hd_FAT32_BPB:
                                <1>       ;cmp  word [esi+BPB_FATSz16], 0
                                <1>       ;ja   short loc_not_a_valid_fat_partition4
                                <1> loc_hd_big_FAT16_BPB:
0000760B B92D000000             <1>       mov   ecx, 45
                                <1>
                                <1> loc_hd_move_FAT_BPB:
00007610 89EF                   <1>       mov   edi, ebp
                                <1>       ;mov  esi, ebx ; Boot sector
00007612 57                     <1>       push  edi
00007613 83C706                 <1>       add   edi, LD_BPB
00007616 F366A5                 <1>       rep   movsw
00007619 5E                     <1>       pop   esi
0000761A 0FB74614               <1>       movzx eax, word [esi+LD_BPB+BPB_RsvdSecCnt]
0000761E 03466C                 <1>       add   eax, [esi+LD_StartSector]
00007621 894660                 <1>       mov   [esi+LD_FATBegin], eax
00007624 807E0303               <1>       cmp   byte [esi+LD_FATType], 3
00007628 7224                   <1>       jb    short loc_set_FAT16_RootDirLoc
                                <1> loc_set_FAT32_RootDirLoc:
0000762A 8B462A                 <1>       mov   eax, [esi+LD_BPB+BPB_FATSz32]
0000762D 0FB65E16               <1>         movzx      ebx, byte [esi+LD_BPB+BPB_NumFATs]
00007631 F7E3                   <1>       mul   ebx
00007633 034660                 <1>       add   eax, [esi+LD_FATBegin]
                                <1> loc_set_FAT32_data_begin:
00007636 894668                 <1>       mov   [esi+LD_DATABegin], eax
00007639 894664                 <1>       mov   [esi+LD_ROOTBegin], eax
                                <1>       ; If Root Directory Cluster <> 2 then
                                <1>       ; change the beginning sector value
                                <1>       ; of the root dir by adding sector offset.
0000763C 8B4632                 <1>       mov   eax, [esi+LD_BPB+BPB_RootClus]
0000763F 83E802                 <1>       sub   eax, 2
00007642 7435                   <1>       jz    short short loc_set_32bit_FAT_total_sectors
                                <1>       ;movzx ebx, byte [esi+LD_BPB+BPB_SecPerClust]
00007644 8A5E13                 <1>       mov   bl, [esi+LD_BPB+BPB_SecPerClust]
00007647 F7E3                   <1>       mul   ebx
00007649 014664                 <1>       add   [esi+LD_ROOTBegin], eax
0000764C EB2B                   <1>       jmp   short loc_set_32bit_FAT_total_sectors
                                <1>       ;
                                <1> loc_set_FAT16_RootDirLoc:
0000764E 0FB64616               <1>       movzx  eax, byte [esi+LD_BPB+BPB_NumFATs]
00007652 0FB7561C               <1>       movzx  edx, word [esi+LD_BPB+BPB_FATSz16]
00007656 F7E2                   <1>       mul   edx
00007658 034660                 <1>       add   eax, [esi+LD_FATBegin]
0000765B 894664                 <1>       mov   [esi+LD_ROOTBegin], eax
                                <1> loc_set_FAT16_data_begin:
0000765E 894668                 <1>       mov   [esi+LD_DATABegin], eax
                                <1>       ;mov  eax, 20h  ; Size of a directory entry
                                <1>       ;;movzx      edx, word [esi+LD_BPB+BPB_RootEntCnt]
                                <1>         ;mov     dx, [esi+LD_BPB+BPB_RootEntCnt]
                                <1>         ;mul edx
                                <1>       ;;mov  ecx, 511
                                <1>       ;mov   cx, 511
                                <1>       ;add   eax, ecx
                                <1>       ;inc   ecx ; 512
                                <1>       ;div   ecx
                                <1>       ; 14/07/2020
00007661 0FB74617               <1>       movzx  eax, word [esi+LD_BPB+BPB_RootEntCnt]
00007665 6683C00F               <1>       add   ax, 15
00007669 66C1E804               <1>       shr   ax, 4 ; / 16 ; (16 entries per sector)
0000766D 014668                 <1>       add   [esi+LD_DATABegin], eax
                                <1>       ;movzx eax, word [esi+LD_BPB+BPB_TotalSec16]
00007670 668B4619               <1>       mov   ax, [esi+LD_BPB+BPB_TotalSec16]
```

```
3868 00007674 6685C0              <1>       test   ax, ax
3869                              <1>       ;jz    short loc_set_32bit_FAT_total_sectors
3870                              <1> ;loc_set_16bit_FAT_total_sectors:
3871                              <1>       ;mov   [esi+LD_TotalSectors], eax
3872                              <1>       ;jmp   short loc_set_hd_FAT_cluster_count
3873                              <1>       ; 14/07/2020
3874 00007677 7503                <1>       jnz    short loc_set_hd_FAT_cluster_count
3875                              <1> loc_set_32bit_FAT_total_sectors:
3876 00007679 8B4626              <1>       mov    eax, [esi+LD_BPB+BPB_TotalSec32]
3877                              <1>       ;mov   [esi+LD_TotalSectors], eax
3878                              <1> loc_set_hd_FAT_cluster_count:
3879 0000767C 894670              <1>       mov    [esi+LD_TotalSectors], eax ; 14/07/2020
3880 0000767F 8B5668              <1>       mov    edx, [esi+LD_DATABegin]
3881 00007682 2B566C              <1>       sub    edx, [esi+LD_StartSector]
3882 00007685 29D0                <1>       sub    eax, edx
3883 00007687 31D2                <1>       xor    edx, edx ; 0
3884 00007689 0FB64E13            <1>         movzx   ecx, byte [esi+LD_BPB+BPB_SecPerClust]
3885 0000768D F7F1                <1>         div  ecx
3886 0000768F 894678              <1>       mov    [esi+LD_Clusters], eax
3887                              <1>       ; Maximum Valid Cluster Number= EAX +1
3888                              <1>       ; with 2 reserved clusters= EAX +2
3889                              <1> loc_set_hd_FAT_fs_free_sectors:
3890                              <1>       ;mov   dword [esi+LD_FreeSectors], 0
3891 00007692 E855010000          <1>       call   get_free_FAT_sectors
3892 00007697 720D                <1>       jc     short loc_validate_hd_FAT_partition_retn
3893 00007699 894674              <1>       mov    [esi+LD_FreeSectors], eax
3894 0000769C C6467E06            <1>       mov    byte [esi+LD_MediaChanged], 6  ; Volume Name Reset
3895                              <1>
3896                              <1>       ; 15/07/2020
3897 000076A0 FE05[54390100]      <1>       inc    byte [Last_DOS_DiskNo] ; > 1
3898                              <1>
3899                              <1>       ;mov   cl, [Last_DOS_DiskNo]
3900                              <1>       ;add   cl, 'A'
3901                              <1>       ;mov   [esi+LD_FS_Name], cl
3902                              <1>
3903                              <1> loc_validate_hd_FAT_partition_retn:
3904 000076A6 C3                  <1>       retn
3905                              <1>
3906                              <1> validate_hd_fs_partition:
3907                              <1>       ; 03/02/2018
3908                              <1>       ; 09/12/2017
3909                              <1>       ; 13/02/2016
3910                              <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
3911                              <1>       ; 29/01/2011
3912                              <1>       ; 23/07/2011
3913                              <1>       ; Input
3914                              <1>       ;   DL = Hard/Fixed Disk Drive Number
3915                              <1>       ;   ESI = PartitionTable offset
3916                              <1>       ;   byte [Last_DOS_DiskNo]
3917                              <1>       ; Output
3918                              <1>       ;  cf=0 -> Validated
3919                              <1>       ;   ESI = Logical dos drv desc. table
3920                              <1>       ;   EBX = Singlix FS boot sector buffer
3921                              <1>       ;   byte [Last_DOS_DiskNo]
3922                              <1>       ;  cf=1 -> Not a valid 'Singlix FS' partition
3923                              <1>       ; EAX, EDX, ECX, EDI -> changed
3924                              <1>
3925                              <1>       ;mov   esi, PartitionTable
3926 000076A7 8A6604              <1>       mov    ah, [esi+ptFileSystemID]
3927 000076AA 80FCA1              <1>       cmp    ah, 0A1h ; SINGLIX FS1 (trfs1) partition
3928 000076AD 7549                <1>       jne    short loc_validate_hd_fs_partition_stc_retn
3929                              <1> loc_set_valid_hd_fs_partition_params:
3930 000076AF FE05[54390100]      <1>       inc    byte [Last_DOS_DiskNo] ; > 1
3931 000076B5 30C0                <1>       xor    al, al ; mov al, 0
3932                              <1>       ;mov   [drv], dl
3933 000076B7 29DB                <1>       sub    ebx, ebx ; 0
3934 000076B9 8A3D[54390100]      <1>       mov    bh, [Last_DOS_DiskNo]
3935 000076BF 81C300010900        <1>       add    ebx, Logical_DOSDisks
3936 000076C5 C6430102            <1>       mov    byte [ebx+LD_DiskType], 2
3937 000076C9 885302              <1>       mov    [ebx+LD_PhyDrvNo], dl
3938                              <1>       ;mov   [ebx+LD_FATType], al ; 0
3939                              <1>       ;mov   [ebx+LD_FSType], ah
3940 000076CC 66894303            <1>       mov    [ebx+LD_FATType], ax
3941                              <1>       ;mov   eax, [esi+ptStartSector]
3942                              <1>       ;mov   [ebx+LD_StartSector], eax
3943                              <1> loc_hd_fs_logical_drv_init:
3944 000076D0 89DD                <1>       mov    ebp, ebx ; 10/01/2016
3945                              <1>       ;mov   dl, [ebx+LD_PhyDrvNo]
3946 000076D2 A0[72860100]        <1>       mov    al, [HD_LBA_yes] ; 10/01/2016
3947 000076D7 884305              <1>       mov    [ebx+LD_LBAYes], al
3948 000076DA 89DE                <1>       mov    esi, ebx
3949 000076DC BB[7E860100]        <1>       mov    ebx, DOSBootSectorBuff ; buffer address
3950 000076E1 08C0                <1>       or     al, al
3951 000076E3 7515                <1>       jnz    short loc_hd_fs_drv_init_load_bs_lba
3952                              <1> loc_hd_fs_drv_init_load_bs_chs:
3953 000076E5 8A7601              <1>       mov    dh, [esi+ptBeginHead]
3954 000076E8 668B4E02            <1>       mov    cx, [esi+ptBeginSector]
3955 000076EC 66B80102            <1>       mov    ax, 0201h ; Read 1 sector
3956                              <1>       ;mov   ebx, DOSBootSectorBuff
3957 000076F0 E86EDBFFFF          <1>       call   int13h
3958 000076F5 7311                <1>       jnc    short loc_hd_drv_fs_boot_validation
3959                              <1> loc_validate_hd_fs_partition_err_retn:
3960 000076F7 C3                  <1>       retn
3961                              <1> loc_validate_hd_fs_partition_stc_retn:
3962 000076F8 F9                  <1>       stc
3963 000076F9 C3                  <1>       retn
3964                              <1> loc_hd_fs_drv_init_load_bs_lba:
3965                              <1>       ; DL = Physical drive number
3966                              <1>       ;mov   esi, ebx
3967 000076FA 8B4E08              <1>       mov    ecx, [esi+ptStartSector] ; sector number
3968                              <1>       ;mov   ebx, DOSBootSectorBuff ; buffer address
3969                              <1>       ; LBA read/write (with private LBA function)
3970                              <1>       ;((Retro UNIX 386 v1 - DISK I/O code by Erdogan Tan))
3971                              <1>       ; dl = physical drive number (0,1, 80h, 81h, 82h, 83h)
3972 000076FD B41B                <1>       mov    ah, 1Bh ; LBA read
```

```
3973 000076FF B001                <1>       mov    al, 1 ; sector count
3974 00007701 E85DDBFFFF          <1>       call   int13h
3975 00007706 72EF                <1>       jc     short loc_validate_hd_fs_partition_err_retn
3976                              <1> loc_hd_drv_fs_boot_validation:
3977                              <1>       ;mov   esi, DOSBootSectorBuff
3978 00007708 89DE                <1>       mov    esi, ebx ; Boot sector buffer
3979 0000770A 6681BEFE01000055AA  <1>       cmp    word [esi+BS_Validation], 0AA55h
3980 00007713 75E3                <1>       jne    short loc_validate_hd_fs_partition_stc_retn
3981                              <1>         ;
3982                              <1>       ;Singlix FS Extensions to TR-DOS (7/6/2009)
3983 00007715 66817E034653        <1>       cmp    word [esi+bs_FS_Identifier], 'FS' ; 03/02/2018
3984 0000771B 75DB                <1>       jne    short loc_validate_hd_fs_partition_stc_retn
3985                              <1>         ;'A1h' check is not necessary
3986                              <1>       ;  if 'FS' check is passed as OK/Yes.
3987 0000771D 807E09A1            <1>       cmp    byte [esi+bs_FS_PartitionID], 0A1h
3988 00007721 75D5                <1>       jne    short loc_validate_hd_fs_partition_stc_retn
3989                              <1>       ;
3990 00007723 89EF                <1>       mov    edi, ebp ; 10/01/2016
3991                              <1>       ;
3992 00007725 8A462D              <1>       mov    al, byte [esi+bs_FS_LBA_Ready]
3993 00007728 884705              <1>       mov    [edi+LD_FS_LBAYes], al
3994                              <1>       ;
3995                              <1>       ; 03/01/2010 CHS -> DOS FAT/BPB compatibility fix
3996 0000772B 8A4608              <1>       mov    al, [esi+bs_FS_MediaAttrib]
3997 0000772E 884706              <1>       mov    byte [edi+LD_FS_MediaAttrib], al
3998                              <1>       ;
3999 00007731 8A460A              <1>       mov    al, [esi+bs_FS_VersionMaj]
4000 00007734 884707              <1>       mov    [edi+LD_FS_VersionMajor], al
4001                              <1>       ;
4002 00007737 668B4606            <1>       mov    ax, [esi+bs_FS_BytesPerSec]
4003 0000773B 66894711            <1>       mov    [edi+LD_FS_BytesPerSec], ax
4004 0000773F 8A462E              <1>       mov    al, [esi+bs_FS_SecPerTrack]
4005 00007742 30E4                <1>       xor    ah, ah ; 09/12/2017
4006 00007744 6689471E            <1>       mov    [edi+LD_FS_SecPerTrack], ax
4007 00007748 8A462F              <1>       mov    al, [esi+bs_FS_Heads]
4008 0000774B 66894720            <1>       mov    [edi+LD_FS_NumHeads], ax
4009                              <1>       ;
4010 0000774F 8B4628              <1>       mov    eax, [esi+bs_FS_UnDelDirD]
4011 00007752 894722              <1>       mov    [edi+LD_FS_UnDelDirD], eax
4012 00007755 8B5618              <1>       mov    edx, [esi+bs_FS_MATLocation]
4013 00007758 89570C              <1>       mov    [edi+LD_FS_MATLocation], edx
4014 0000775B 8B461C              <1>       mov    eax, [esi+bs_FS_RootDirD]
4015 0000775E 894708              <1>       mov    [edi+LD_FS_RootDirD], eax
4016 00007761 8B460C              <1>       mov    eax, [esi+bs_FS_BeginSector]
4017 00007764 89476C              <1>       mov    [edi+LD_FS_BeginSector], eax
4018 00007767 8B4710              <1>       mov    eax, [edi+bs_FS_VolumeSize]
4019 0000776A 894770              <1>       mov    [edi+LD_FS_VolumeSize], eax
4020                              <1>       ;
4021 0000776D 89D0                <1>       mov    eax, edx ; [edi+LD_FS_MATLocation]
4022 0000776F 03476C              <1>       add    eax, [edi+LD_FS_BeginSector]
4023 00007772 89FE                <1>       mov    esi, edi
4024                              <1> mread_hd_fs_MAT_sector:
4025                              <1>        ;mov  ebx, DOSBootSectorBuff
4026 00007774 B901000000          <1>       mov    ecx, 1
4027 00007779 E81DAE0000          <1>       call   disk_read
4028 0000777E 7248                <1>       jc     short loc_validate_hd_fs_partition_retn
4029                              <1>       ; EDI will not be changed
4030 00007780 89DE                <1>       mov    esi, ebx
4031                              <1> use_hdfs_mat_sector_params:
4032 00007782 8B460C              <1>       mov    eax, [esi+FS_MAT_DATLocation]
4033 00007785 894714              <1>       mov    [edi+LD_FS_DATLocation], eax
4034 00007788 8B4610              <1>       mov    eax, [esi+FS_MAT_DATScount]
4035 0000778B 894718              <1>       mov    [edi+LD_FS_DATSectors], eax
4036 0000778E 8B4614              <1>       mov    eax, [esi+FS_MAT_FreeSectors]
4037 00007791 894774              <1>         mov     [edi+LD_FS_FreeSectors], eax
4038 00007794 8B4618              <1>       mov    eax, [esi+FS_MAT_FirstFreeSector]
4039 00007797 894778              <1>       mov    [edi+LD_FS_FirstFreeSector], eax
4040 0000779A 8B4708              <1>       mov    eax, [edi+LD_FS_RootDirD]
4041 0000779D 03476C              <1>       add    eax, [edi+LD_FS_BeginSector]
4042 000077A0 89FE                <1>       mov    esi, edi
4043                              <1> read_hd_fs_RDT_sector:
4044 000077A2 BB[7E860100]        <1>       mov    ebx, DOSBootSectorBuff
4045                              <1>       ;mov   ecx, 1
4046 000077A7 B101                <1>       mov    cl, 1
4047 000077A9 E8EDAD0000          <1>       call   disk_read
4048 000077AE 7218                <1>       jc     short loc_validate_hd_fs_partition_retn
4049                              <1>       ; EDI will not be changed
4050 000077B0 89DE                <1>       mov    esi, ebx
4051                              <1> use_hdfs_RDT_sector_params:
4052 000077B2 8B461C              <1>       mov    eax, [esi+FS_RDT_VolumeSerialNo]
4053 000077B5 894728              <1>       mov    [edi+LD_FS_VolumeSerial], eax
4054 000077B8 57                  <1>       push   edi
4055                              <1>       ;mov   ecx, 16
4056 000077B9 B110                <1>       mov    cl, 16
4057 000077BB 83C640              <1>       add    esi, FS_RDT_VolumeName
4058 000077BE 83C72C              <1>       add    edi, LD_FS_VolumeName
4059 000077C1 F3A5                <1>       rep    movsd ; 64 bytes
4060 000077C3 5E                  <1>       pop    esi
4061                              <1>              ; Volume Name Reset
4062 000077C4 C6467E06            <1>         mov     byte [esi+LD_FS_MediaChanged], 6
4063                              <1>       ;
4064                              <1>         ;mov cl, [Last_DOS_DiskNo]
4065                              <1>       ;add   cl, 'A'
4066                              <1>       ;mov   [esi+LD_FS_Name], cl
4067                              <1>
4068                              <1> loc_validate_hd_fs_partition_retn:
4069 000077C8 C3                  <1>       retn
4070                              <1>
4071                              <1> load_masterboot:
4072                              <1>       ; 14/07/2020 (Reset function has been removed)
4073                              <1>       ;
4074                              <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
4075                              <1>       ; 2005 - 2011
4076                              <1>       ; input -> DL = drive number
4077                              <1> ;     mov    ah, 0Dh ; Alternate disk reset
```

```
4078                                 <1> ;       call    int13h
4079                                 <1> ;       jnc     short pass_reset_error
4080                                 <1> ;harddisk_error:
4081                                 <1> ;       retn
4082                                 <1> ;pass_reset_error:
4083 000077C9 BB[6E830100]           <1>         mov     ebx, MasterBootBuff
4084 000077CE 66B80102               <1>         mov     ax, 0201h
4085 000077D2 66B90100               <1>         mov     cx, 1
4086 000077D6 30F6                   <1>         xor     dh, dh
4087 000077D8 E886DAFFFF             <1>         call    int13h
4088 000077DD 720C                   <1>         jc      short harddisk_error
4089                                 <1>         ;
4090 000077DF 66813D[6C850100]55-    <1>         cmp     word [MBIDCode], 0AA55h
4090 000077E7 AA                     <1>
4091 000077E8 7401                   <1>         je      short load_masterboot_ok
4092 000077EA F9                     <1>         stc
4093                                 <1> harddisk_error:
4094                                 <1> load_masterboot_ok:
4095 000077EB C3                     <1>         retn
4096                                 <1>
4097                                 <1> get_free_FAT_sectors:
4098                                 <1>         ; 21/12/2017
4099                                 <1>         ; 29/02/2016
4100                                 <1>         ; 13/02/2016
4101                                 <1>         ; 04/02/2016
4102                                 <1>         ; 07/01/2016 (TRDOS 386 = TRDOS v2.0)
4103                                 <1>         ; 11/07/2010
4104                                 <1>         ; 21/06/2009
4105                                 <1>         ; INPUT: ESI = Logical DOS Drive Description Table address
4106                                 <1>         ; OUTPUT: STC => Error
4107                                 <1>           ;    cf = 0 and EAX = Free FAT sectors
4108                                 <1>         ; Also, related parameters and FAT buffer will be reset and updated
4109                                 <1>
4110 000077EC 31C0                   <1>         xor     eax, eax
4111                                 <1>         ;mov   [esi+LD_FreeSectors], eax ; Reset
4112                                 <1>
4113 000077EE 807E0302               <1>           cmp     byte [esi+LD_FATType], 2
4114 000077F2 7654                   <1>         jna     short loc_gfc_get_fat_free_clusters
4115                                 <1>
4116                                 <1>         ; 29/02/2016
4117 000077F4 48                     <1>         dec     eax ; 0FFFFFFFFh
4118 000077F5 89463A                 <1>         mov     [esi+LD_BPB+BPB_Reserved], eax ; Free cluster count (reset)
4119 000077F8 89463E                 <1>         mov     [esi+LD_BPB+BPB_Reserved+4], eax ; First Free Cluster (reset)
4120 000077FB 40                     <1>         inc     eax ; 0
4121                                 <1>         ;
4122 000077FC 668B4636               <1>         mov     ax, [esi+LD_BPB+BPB_FSInfo]
4123 00007800 03466C                 <1>         add     eax, [esi+LD_StartSector]
4124                                 <1>
4125 00007803 BB[7E860100]           <1>         mov     ebx, DOSBootSectorBuff
4126 00007808 B901000000             <1>         mov     ecx, 1
4127 0000780D E889AD0000             <1>         call    disk_read
4128 00007812 7301                   <1>         jnc     short loc_gfc_check_fsinfo_signs
4129                                 <1> retn_gfc_get_fsinfo_sec:
4130 00007814 C3                     <1>         retn
4131                                 <1>
4132                                 <1> loc_gfc_check_fsinfo_signs:
4133 00007815 BB[7E860100]           <1>         mov     ebx, DOSBootSectorBuff ; 13/02/2016
4134 0000781A 813B52526141           <1>           cmp     dword [ebx], 41615252h
4135 00007820 7524                   <1>         jne     short retn_gfc_get_fsinfo_stc
4136                                 <1>         ;add    ebx, 484
4137                                 <1>         ;cmp    dword [ebx], 61417272h
4138 00007822 81BBE4010000727241-    <1>         cmp     dword [ebx+484], 61417272h
4138 0000782B 61                     <1>
4139 0000782C 7518                   <1>         jne     short retn_gfc_get_fsinfo_stc
4140                                 <1>         ;add    ebx, 4
4141                                 <1>         ;mov    eax, [ebx]
4142 0000782E 8B83E8010000           <1>         mov     eax, [ebx+488]
4143                                 <1>         ; 29/02/2016
4144 00007834 89463A                 <1>         mov     [esi+LD_BPB+BPB_Reserved], eax ; Free cluster count
4145 00007837 8B93EC010000           <1>         mov     edx,  [ebx+492]
4146 0000783D 89463E                 <1>         mov     [esi+LD_BPB+BPB_Reserved+4], eax ; First Free Cluster
4147                                 <1>         ; 21/12/2017
4148 00007840 89C3                   <1>         mov     ebx, eax ; (initial value = 0FFFFFFFFh)
4149 00007842 43                     <1>         inc     ebx ; 0FFFFFFFFh -> 0
4150 00007843 7513                   <1>         jnz     short short retn_from_get_free_fat32_clusters
4151 00007845 C3                     <1>         retn
4152                                 <1>
4153                                 <1> retn_gfc_get_fsinfo_stc:
4154 00007846 F9                     <1>         stc
4155 00007847 C3                     <1>         retn
4156                                 <1>
4157                                 <1> loc_gfc_get_fat_free_clusters:
4158                                 <1>         ;mov   eax, 2
4159 00007848 B002                   <1>         mov     al, 2
4160                                 <1>         ;mov   [FAT_CurrentCluster], eax
4161                                 <1> loc_gfc_loop_get_next_cluster:
4162 0000784A E8EB4F0000             <1>         call    get_next_cluster
4163 0000784F 730E                   <1>         jnc     short loc_gfc_free_fat_clusters_cont
4164 00007851 21C0                   <1>         and     eax, eax
4165 00007853 7411                   <1>         jz      short loc_gfc_pass_inc_free_cluster_count
4166                                 <1>
4167                                 <1> retn_from_get_free_fat_clusters:
4168 00007855 8B4674                 <1>         mov     eax, [esi+LD_FreeSectors] ; Free clusters !
4169                                 <1> retn_from_get_free_fat32_clusters:
4170 00007858 0FB65E13               <1>           movzx     ebx, byte [esi+LD_BPB+BPB_SecPerClust]
4171 0000785C F7E3                   <1>               mul     ebx
4172                                 <1>         ;mov   [esi+LD_FreeSectors], eax ; Free sectors
4173                                 <1> retn_get_free_sectors_calc:
4174 0000785E C3                     <1>         retn
4175                                 <1>
4176                                 <1> loc_gfc_free_fat_clusters_cont:
4177 0000785F 09C0                   <1>         or      eax, eax
4178 00007861 7503                   <1>         jnz     short loc_gfc_pass_inc_free_cluster_count
4179 00007863 FF4674                 <1>         inc     dword [esi+LD_FreeSectors] ; Free clusters !
4180                                 <1>
```

```
                        <1> loc_gfc_pass_inc_free_cluster_count:
                        <1>       ;mov   eax, [FAT_CurrentCluster]
00007866 89C8           <1>       mov    eax, ecx ; [FAT_CurrentCluster]
00007868 3B4678         <1>       cmp    eax, [esi+LD_Clusters]
0000786B 77E8           <1>       ja     short retn_from_get_free_fat_clusters
0000786D 40             <1>       inc    eax
                        <1>       ;mov   [FAT_CurrentCluster], eax
0000786E EBDA           <1>       jmp    short loc_gfc_loop_get_next_cluster
                        <1>
                        <1> floppy_drv_init:
                        <1>       ; 09/12/2017
                        <1>       ; 06/07/2016
                        <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>       ; 24/07/2011
                        <1>       ; 04/07/2009
                        <1>       ; INPUT ->
                        <1>       ;      DL = Drive number (0,1)
                        <1>       ; OUTPUT ->
                        <1>       ;      BL = drive name
                        <1>       ;      BH = drive number
                        <1>       ;      ESI = Logical DOS drv description table
                        <1>       ;      EAX = Volume serial number
                        <1>
00007870 BE[EE680000]   <1>       mov    esi, fd0_type ; 10/01/2016
00007875 BF00010900     <1>       mov    edi, Logical_DOSDisks
0000787A 08D2           <1>       or     dl, dl
0000787C 7407           <1>       jz     short loc_drv_init_fd0_fd1
0000787E 81C700010000   <1>       add    edi, 100h
00007884 46             <1>       inc    esi ; fd1_type ; 10/01/2016
                        <1> loc_drv_init_fd0_fd1:
00007885 C6477E00       <1>       mov    byte [edi+LD_MediaChanged], 0
00007889 803E01         <1>       cmp    byte [esi], 1 ; type (>0 if it is existing)
                        <1>              ; 4 = 1.44 MB, 80 track, 3 1/2"
0000788C 7221           <1>       jb     short read_fd_boot_sector_retn
0000788E 885702         <1>       mov    [edi+LD_PhyDrvNo], dl
                        <1> read_fd_boot_sector:
00007891 30F6           <1>       xor    dh, dh
00007893 B904000000     <1>       mov    ecx, 4 ; Retry Count
                        <1> read_fd_boot_sector_again:
00007898 51             <1>       push   ecx
                        <1>       ;mov   cx, 1
00007899 B101           <1>       mov    cl, 1
0000789B 66B80102       <1>       mov    ax, 0201h ; Read 1 sector
0000789F BB[7E860100]   <1>       mov    ebx, DOSBootSectorBuff
000078A4 E8BAD9FFFF     <1>       call   int13h
000078A9 59             <1>       pop    ecx
000078AA 7304           <1>       jnc    short use_fd_boot_sector_params
000078AC E2EA           <1>       loop   read_fd_boot_sector_again
                        <1>
                        <1> read_fd_boot_sector_stc_retn:
000078AE F9             <1>       stc
                        <1> read_fd_boot_sector_retn:
000078AF C3             <1>       retn
                        <1>
                        <1> use_fd_boot_sector_params:
                        <1>       ;mov   esi, DOSBootSectorBuff
000078B0 89DE           <1>       mov    esi, ebx
000078B2 6681BEFE01000055AA <1>   cmp    word [esi+BS_Validation], 0AA55h
000078BB 75F1           <1>       jne    short read_fd_boot_sector_stc_retn
000078BD 66817E035346   <1>         cmp     word [esi+bs_FS_Identifier], 'SF'
000078C3 0F85A2000000   <1>         jne     use_fd_fatfs_boot_sector_params
                        <1>       ;
000078C9 8A462D         <1>       mov    al, [esi+bs_FS_LBA_Ready]
000078CC 884705         <1>       mov    [edi+LD_FS_LBAYes], al
                        <1>       ;
                        <1>       ; 03/01/2010 CHS -> DOS FAT/BPB compatibility fix
000078CF 8A4608         <1>       mov    al, [esi+bs_FS_MediaAttrib]
000078D2 884706         <1>       mov    [edi+LD_FS_MediaAttrib], al
                        <1>       ;
000078D5 8A460A         <1>         mov  al, [esi+bs_FS_VersionMaj]
000078D8 884707         <1>       mov    byte [edi+LD_FS_VersionMajor], al
000078DB 668B4606       <1>       mov    ax, [esi+bs_FS_BytesPerSec]
000078DF 66894711       <1>       mov    [edi+LD_FS_BytesPerSec], ax
000078E3 8A462E         <1>       mov    al, [esi+bs_FS_SecPerTrack]
000078E6 28E4           <1>       sub    ah, ah ; 09/12/2017
000078E8 6689471E       <1>       mov    [edi+LD_FS_SecPerTrack], ax
000078EC 8A462F         <1>       mov    al, [esi+bs_FS_Heads]
000078EF 66894720       <1>       mov    [edi+LD_FS_NumHeads], ax
                        <1>       ;
000078F3 8B4628         <1>       mov    eax, [esi+bs_FS_UnDelDirD]
000078F6 894722         <1>       mov    [edi+LD_FS_UnDelDirD], eax
000078F9 8B4618         <1>       mov    eax, [esi+bs_FS_MATLocation]
000078FC 89470C         <1>       mov    [edi+LD_FS_MATLocation], eax
000078FF 8B461C         <1>       mov    eax, [esi+bs_FS_RootDirD]
00007902 894708         <1>       mov    [edi+LD_FS_RootDirD], eax
00007905 8B460C         <1>       mov    eax, [esi+bs_FS_BeginSector]
00007908 89476C         <1>       mov    [edi+LD_FS_BeginSector], eax
0000790B 8B4610         <1>       mov    eax, [esi+bs_FS_VolumeSize]
0000790E 894770         <1>       mov    [edi+LD_FS_VolumeSize], eax
                        <1>       ;
00007911 89FE           <1>       mov    esi, edi
00007913 8B460C         <1>       mov    eax, [esi+LD_FS_MATLocation]
                        <1>       ;add   eax, [edi+LD_FS_BeginSector]
                        <1> read_fd_MAT_sector_again:
                        <1>       ;mov   ebx, DOSBootSectorBuff
                        <1>       ;mov   ecx, 1
00007916 B101           <1>       mov    cl, 1
00007918 E884AC0000     <1>       call   chs_read
0000791D 89DE           <1>       mov    esi, ebx
0000791F 7301           <1>       jnc    short use_fdfs_mat_sector_params
                        <1>       ;jmp   short read_fd_boot_sector_retn
00007921 C3             <1>       retn
                        <1> use_fdfs_mat_sector_params:
00007922 8B460C         <1>       mov    eax, [esi+FS_MAT_DATLocation]
00007925 894714         <1>       mov    [edi+LD_FS_DATLocation], eax
```

```
4286 00007928 8B4610            <1>       mov    eax, [esi+FS_MAT_DATScount]
4287 0000792B 894718            <1>       mov    [edi+LD_FS_DATSectors], eax
4288 0000792E 8B4714            <1>       mov    eax, [edi+FS_MAT_FreeSectors]
4289 00007931 894774            <1>       mov    [edi+LD_FS_FreeSectors], eax
4290 00007934 8B4618            <1>       mov    eax, [esi+FS_MAT_FirstFreeSector]
4291 00007937 894778            <1>       mov    [edi+LD_FS_FirstFreeSector], eax
4292                            <1>       ;
4293 0000793A 89FE              <1>       mov    esi, edi
4294 0000793C 8B4608            <1>       mov    eax, [esi+LD_FS_RootDirD]
4295                            <1> read_fd_RDT_sector_again:
4296                            <1>       ;mov   ebx, DOSBootSectorBuff
4297                            <1>       ;mov   cx, 1
4298 0000793F B101              <1>       mov    cl, 1
4299 00007941 E85BAC0000        <1>       call   chs_read
4300 00007946 89DE              <1>       mov    esi, ebx
4301 00007948 7220              <1>       jc     short read_fd_RDT_sector_retn
4302                            <1> use_fdfs_RDT_sector_params:
4303 0000794A 8B461C            <1>       mov    eax, [esi+FS_RDT_VolumeSerialNo]
4304 0000794D 894728            <1>       mov    [edi+LD_FS_VolumeSerial], eax
4305 00007950 57                <1>       push   edi
4306                            <1>       ;mov   ecx, 16
4307 00007951 B110              <1>       mov    cl, 16
4308 00007953 83C640            <1>       add    esi, FS_RDT_VolumeName
4309 00007956 83C72C            <1>       add    edi, LD_FS_VolumeName
4310 00007959 F3A5              <1>       rep    movsd ; 64 bytes
4311 0000795B 5E                <1>       pop    esi
4312 0000795C C6460300          <1>       mov    byte [esi+LD_FATType], 0
4313 00007960 C64604A1          <1>       mov    byte [esi+LD_FSType], 0A1h
4314 00007964 E9A5000000        <1>         jmp     loc_cont_use_fd_boot_sector_params
4315                            <1>
4316                            <1> read_fd_RDT_sector_stc_retn:
4317 00007969 F9                <1>       stc
4318                            <1> read_fd_RDT_sector_retn:
4319 0000796A C3                <1>       retn
4320                            <1>
4321                            <1> use_fd_fatfs_boot_sector_params:
4322 0000796B 807E2629          <1>       cmp    byte [esi+BS_BootSig], 29h
4323 0000796F 75F8              <1>       jne    short read_fd_RDT_sector_stc_retn
4324 00007971 807E15F0          <1>       cmp    byte [esi+BPB_Media], 0F0h
4325 00007975 72F3              <1>       jb     short read_fd_RDT_sector_retn
4326 00007977 57                <1>       push   edi
4327 00007978 83C706            <1>       add    edi, LD_BPB
4328                            <1>       ;mov   ecx, 16
4329 0000797B B110              <1>       mov    cl, 16
4330 0000797D F3A5              <1>       rep    movsd ; 64 bytes
4331 0000797F 5E                <1>       pop    esi
4332 00007980 31C0              <1>       xor    eax, eax
4333 00007982 89466C            <1>       mov    [esi+LD_StartSector], eax ; 0
4334 00007985 668B461C          <1>       mov    ax, [esi+LD_BPB+BPB_FATSz16]
4335 00007989 8A4E16            <1>       mov    cl, [esi+LD_BPB+BPB_NumFATs]
4336 0000798C F7E1              <1>       mul    ecx
4337                            <1>       ; edx = 0 !
4338 0000798E 668B5614          <1>       mov    dx, [esi+LD_BPB+BPB_RsvdSecCnt]
4339 00007992 66895660          <1>       mov    [esi+LD_FATBegin], dx
4340                            <1>       ;add   eax, edx
4341 00007996 6601D0            <1>       add    ax, dx
4342 00007999 894664            <1>       mov    [esi+LD_ROOTBegin], eax
4343 0000799C 894668            <1>       mov    [esi+LD_DATABegin], eax
4344 0000799F 668B5617          <1>       mov    dx, [esi+LD_BPB+BPB_RootEntCnt]
4345                            <1>       ;;shl  edx, 5 ; * 32 (Size of a directory entry)
4346                            <1>       ;shl   dx, 5
4347                            <1>       ;;add  edx, 511
4348                            <1>       ;add   dx, 511
4349                            <1>       ;;shr  edx, 9 ; edx = ((edx*32)+511) / 512
4350                            <1>       ;shr   dx, 9
4351 000079A3 6683C20F          <1>       add    dx, 15 ; 06/07/2016 (+(512/32)-1)
4352 000079A7 66C1EA04          <1>       shr    dx, 4 ; / 16 (==16 entries per sector)
4353 000079AB 015668            <1>       add    [esi+LD_DATABegin], edx ; + rd sectors
4354                            <1>       ;movzx eax, word [esi+LD_BPB+BPB_TotalSec16]
4355 000079AE 668B4619          <1>       mov    ax, [esi+LD_BPB+BPB_TotalSec16]
4356 000079B2 894670            <1>       mov    [esi+LD_TotalSectors], eax
4357 000079B5 2B4668            <1>       sub    eax, [esi+LD_DATABegin]
4358                            <1>       ;movzx ecx, byte [esi+LD_BPB+BPB_SecPerClust]
4359 000079B8 8A4E13            <1>       mov    cl, [esi+LD_BPB+BPB_SecPerClust]
4360 000079BB 80F901            <1>       cmp    cl, 1
4361 000079BE 7605              <1>       jna    short save_fd_fatfs_cluster_count
4362                            <1>       ;sub   edx, edx
4363 000079C0 6629D2            <1>       sub    dx, dx ; 0
4364                            <1>       ;sub   dl, dl ; 06/07/2016
4365 000079C3 F7F1              <1>       div    ecx
4366                            <1> save_fd_fatfs_cluster_count:
4367 000079C5 894678            <1>       mov    [esi+LD_Clusters], eax
4368                            <1>
4369                            <1>       ; Maximum Valid Cluster Number = EAX +1
4370                            <1>       ; with 2 reserved clusters= EAX +2
4371                            <1>
4372                            <1> reset_FAT_buffer_decriptors:
4373 000079C8 29C0              <1>       sub    eax, eax ; 0
4374 000079CA A2[82880100]      <1>       mov    [FAT_BuffValidData], al ; 0
4375 000079CF A2[83880100]      <1>       mov    [FAT_BuffDrvName], al ; 0
4376 000079D4 A3[86880100]      <1>       mov    [FAT_BuffSector], eax ; 0
4377                            <1>
4378                            <1> read_fd_FAT_sectors:
4379 000079D9 BB001C0900        <1>       mov    ebx, FAT_Buffer
4380 000079DE 668B4614          <1>       mov    ax, [esi+LD_BPB+BPB_RsvdSecCnt]
4381                            <1>       ;mov   ecx, 3
4382 000079E2 B103              <1>       mov    cl, 3 ; 3 sectors
4383 000079E4 E8B8AB0000        <1>       call   chs_read
4384 000079E9 7240              <1>       jc     short read_fd_FAT_sectors_retn
4385                            <1> use_fd_FAT_sectors:
4386 000079EB 8A4602            <1>       mov    al, [esi+LD_PhyDrvNo]
4387 000079EE 0441              <1>       add    al, 'A'
4388 000079F0 A2[83880100]      <1>       mov    [FAT_BuffDrvName], al
4389 000079F5 C605[82880100]01  <1>       mov    byte [FAT_BuffValidData], 1
4390 000079FC E82B000000        <1>       call   fd_init_calculate_free_clusters
```

```
4391 00007A01 7228              <1>       jc     short read_fd_FAT_sectors_retn
4392                            <1>
4393                            <1> loc_use_fd_boot_sector_params_FAT:
4394 00007A03 C6460301          <1>       mov    byte [esi+LD_FATType], 1 ; FAT 12
4395 00007A07 C6460401          <1>       mov    byte [esi+LD_FSType], 1
4396 00007A0B 8B462D            <1>         mov     eax, [esi+LD_BPB+VolumeID]
4397                            <1> loc_cont_use_fd_boot_sector_params:
4398 00007A0E 8A7E02            <1>       mov    bh, [esi+LD_PhyDrvNo]
4399 00007A11 887E7D            <1>       mov    [esi+LD_DParamEntry], bh
4400 00007A14 88FB              <1>       mov    bl, bh
4401 00007A16 80C341            <1>       add    bl, 'A'
4402 00007A19 881E              <1>       mov    byte [esi+LD_Name], bl
4403 00007A1B C6460101          <1>       mov    byte [esi+LD_DiskType], 1
4404 00007A1F C6460500          <1>       mov    byte [esi+LD_LBAYes], 0
4405 00007A23 C6467C00          <1>       mov    byte [esi+LD_PartitionEntry], 0
4406 00007A27 C6467E06          <1>       mov    byte [esi+LD_MediaChanged], 6 ; Volume Name Reset
4407                            <1>
4408                            <1> read_fd_FAT_sectors_retn:
4409 00007A2B C3                <1>       retn
4410                            <1>
4411                            <1> fd_init_calculate_free_clusters:
4412                            <1>       ; 09/12/2017
4413                            <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
4414                            <1>       ; 04/07/2009
4415                            <1>       ; INPUT ->
4416                            <1>       ;     ESI = Logical DOS drive description table address
4417                            <1>       ; OUTPUT ->
4418                            <1>       ;    [ESI+LD_FreeSectors] will be set
4419                            <1>
4420 00007A2C 29C0              <1>       sub    eax, eax
4421 00007A2E 894674            <1>       mov    [esi+LD_FreeSectors], eax ; 0
4422 00007A31 B002              <1>       mov    al, 2 ; eax = 2
4423                            <1>
4424                            <1> fd_init_loop_get_next_cluster:
4425 00007A33 E830000000        <1>       call   fd_init_get_next_cluster
4426 00007A38 722D              <1>       jc     short fd_init_calculate_free_clusters_retn
4427                            <1>
4428                            <1> fd_init_free_fat_clusters:
4429                            <1>       ;cmp   eax, 0
4430                            <1>       ;ja    short fd_init_pass_inc_free_cluster_count
4431                            <1>       ;and   eax, eax
4432                            <1>       ;jnz   short fd_init_pass_inc_free_cluster_count
4433 00007A3A 6621C0            <1>       and    ax, ax
4434 00007A3D 7504              <1>       jnz    short fd_init_pass_inc_free_cluster_count
4435                            <1>       ;inc   dword [esi+LD_FreeSectors]
4436 00007A3F 66FF4674          <1>         inc  word [esi+LD_FreeSectors]
4437                            <1>
4438                            <1> fd_init_pass_inc_free_cluster_count:
4439                            <1>       ;mov   eax, [FAT_CurrentCluster]
4440 00007A43 66A1[7E880100]    <1>       mov    ax, [FAT_CurrentCluster]
4441                            <1>       ;cmp   eax, [esi+LD_Clusters]
4442 00007A49 663B4678          <1>       cmp    ax, [esi+LD_Clusters]
4443 00007A4D 7704              <1>       ja     short short retn_from_fd_init_calculate_free_clusters
4444                            <1>       ;inc   eax
4445 00007A4F 6640              <1>       inc    ax
4446 00007A51 EBE0              <1>       jmp    short fd_init_loop_get_next_cluster
4447                            <1>
4448                            <1> retn_from_fd_init_calculate_free_clusters:
4449 00007A53 8A4613            <1>       mov    al, [esi+LD_BPB+BPB_SecPerClust]
4450 00007A56 3C01              <1>       cmp    al, 1
4451 00007A58 760D              <1>       jna    short fd_init_calculate_free_clusters_retn
4452                            <1>       ;movzx eax, al
4453 00007A5A 30E4              <1>       xor    ah, ah ; 09/12/2017
4454                            <1>       ;mov   ecx, [esi+LD_FreeSectors]
4455 00007A5C 668B4E74          <1>       mov    cx, [esi+LD_FreeSectors] ; Count of free clusters
4456                            <1>       ;mul   ecx
4457 00007A60 66F7E1            <1>       mul    cx
4458                            <1>       ;mov   [esi+LD_FreeSectors], eax
4459 00007A63 66894674          <1>       mov    [esi+LD_FreeSectors], ax
4460                            <1> fd_init_calculate_free_clusters_retn:
4461 00007A67 C3                <1>       retn
4462                            <1>
4463                            <1> fd_init_get_next_cluster:
4464                            <1>       ; 04/02/2016
4465                            <1>       ; 02/02/2016
4466                            <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
4467                            <1>       ; 04/07/2009
4468                            <1>       ; INPUT ->
4469                            <1>       ;    EAX = Current cluster
4470                            <1>       ;    ESI = Logical DOS drive description table address
4471                            <1>       ;    EDX = 0
4472                            <1>       ; OUTPUT ->
4473                            <1>       ;    EAX = Next cluster
4474                            <1>
4475 00007A68 A3[7E880100]     <1>       mov    [FAT_CurrentCluster], eax
4476                            <1> fd_init_get_next_cluster_readnext:
4477 00007A6D 29D2              <1>       sub    edx, edx ; 0
4478 00007A6F BB00040000        <1>       mov    ebx, 1024 ; 400h
4479 00007A74 F7F3              <1>       div    ebx
4480                            <1>       ; EAX = Count of 3 FAT sectors
4481                            <1>       ; EDX = Buffer entry index
4482 00007A76 89C1              <1>       mov    ecx, eax
4483                            <1>       ;mov   eax, 3
4484 00007A78 B003              <1>       mov    al, 3
4485 00007A7A F7E2              <1>       mul    edx ; Multiply by 3
4486 00007A7C 66D1E8            <1>       shr    ax, 1 ; Divide by 2
4487 00007A7F 89C3              <1>       mov    ebx, eax ; Buffer byte offset
4488 00007A81 81C3001C0900      <1>       add    ebx, FAT_Buffer
4489 00007A87 89C8              <1>       mov    eax, ecx
4490                            <1>       ;mov   edx, 3
4491 00007A89 66BA0300          <1>       mov    dx, 3
4492 00007A8D F7E2              <1>       mul    edx
4493                            <1>       ; EAX = FAT Beginning Sector
4494                            <1>       ; EDX = 0
4495 00007A8F 8A0E              <1>       mov    cl, [esi+LD_Name]
```

```
4496                                    <1>       ;cmp   byte [FAT_BuffValidData], 0
4497                                    <1>       ;jna   short fd_init_load_FAT_sectors0
4498 00007A91 3A0D[83880100]           <1>       cmp    cl, [FAT_BuffDrvName]
4499 00007A97 751E                     <1>       jne    short fd_init_load_FAT_sectors0
4500 00007A99 3B05[86880100]           <1>       cmp    eax, [FAT_BuffSector]
4501 00007A9F 751C                     <1>       jne    short fd_init_load_FAT_sectors1
4502                                    <1>       ;mov   eax, [FAT_CurrentCluster]
4503 00007AA1 A0[7E880100]             <1>       mov    al, [FAT_CurrentCluster]
4504                                    <1>       ;shr   eax, 1
4505 00007AA6 D0E8                     <1>       shr    al, 1
4506 00007AA8 668B03                   <1>       mov    ax, [ebx]
4507 00007AAB 7306                     <1>       jnc    short fd_init_gnc_even
4508 00007AAD 66C1E804                 <1>       shr    ax, 4
4509                                    <1> fd_init_gnc_clc_retn:
4510 00007AB1 F8                       <1>       clc
4511 00007AB2 C3                       <1>       retn
4512                                    <1>
4513                                    <1> fd_init_gnc_even:
4514 00007AB3 80E40F                   <1>       and    ah, 0Fh
4515 00007AB6 C3                       <1>       retn
4516                                    <1>
4517                                    <1> fd_init_load_FAT_sectors0:
4518 00007AB7 880D[83880100]           <1>       mov    [FAT_BuffDrvName], cl
4519                                    <1> fd_init_load_FAT_sectors1:
4520 00007ABD C605[82880100]00         <1>       mov    byte [FAT_BuffValidData], 0
4521 00007AC4 A3[86880100]             <1>       mov    [FAT_BuffSector], eax
4522 00007AC9 034660                   <1>       add    eax, [esi+LD_FATBegin]
4523 00007ACC BB001C0900               <1>       mov    ebx, FAT_Buffer
4524                                    <1>       ;movzx ecx, word [esi+LD_BPB+BPB_FATSz16]
4525 00007AD1 668B4E1C                 <1>       mov    cx, [esi+LD_BPB+BPB_FATSz16]
4526 00007AD5 662B0D[86880100]         <1>       sub    cx, [FAT_BuffSector]
4527                                    <1>         ;cmp ecx, 3
4528 00007ADC 6683F903                 <1>       cmp    cx, 3
4529 00007AE0 7605                     <1>       jna    short fdinit_pass_fix_sector_count_3
4530                                    <1>       ;mov   ecx, 3
4531 00007AE2 B903000000               <1>       mov    ecx, 3
4532                                    <1> fdinit_pass_fix_sector_count_3:
4533 00007AE7 E8B5AA0000               <1>       call   chs_read
4534 00007AEC 730D                     <1>       jnc    short fd_init_FAT_sectors_no_load_error
4535 00007AEE C605[82880100]00         <1>       mov    byte [FAT_BuffValidData], 0
4536                                    <1>              ; Drv not ready or read Error !
4537 00007AF5 B80F000000               <1>       mov    eax, ERR_DRV_NOT_RDY ; 15
4538                                    <1>       ;xor   edx, edx
4539 00007AFA C3                       <1>       retn
4540                                    <1>
4541                                    <1> fd_init_FAT_sectors_no_load_error:
4542 00007AFB C605[82880100]01         <1>       mov    byte [FAT_BuffValidData], 1
4543 00007B02 A1[7E880100]             <1>       mov    eax, [FAT_CurrentCluster]
4544 00007B07 E961FFFFFF               <1>         jmp     fd_init_get_next_cluster_readnext
4545                                    <1>
4546                                    <1> get_FAT_volume_name:
4547                                    <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
4548                                    <1>       ; 12/09/2009
4549                                    <1>       ; INPUT ->
4550                                    <1>       ;      BH = Logical DOS drive number (0,1,2,3,4 ...)
4551                                    <1>       ;       BL = 0
4552                                    <1>       ; OUTPUT ->
4553                                    <1>       ;      CF = 0 -> ESI = Volume name address
4554                                    <1>       ;      CF = 1 -> Root volume name not found
4555                                    <1>
4556                                    <1>       ;mov   ah, 0FFh
4557                                    <1>       ;mov   al, [Last_Dos_DiskNo]
4558                                    <1>       ;cmp   al, bh
4559                                    <1>       ;jb     short loc_gfvn_dir_load_err
4560                                    <1>
4561 00007B0C 89DE                     <1>       mov    esi, ebx
4562 00007B0E 81E600FF0000             <1>       and    esi, 0FF00h ; esi = bh
4563 00007B14 81C600010900             <1>       add    esi, Logical_DOSDisks
4564 00007B1A 8A06                     <1>       mov     al, [esi+LD_Name]
4565 00007B1C 8A6603                   <1>       mov     ah, [esi+LD_FATType]
4566 00007B1F 80FC01                   <1>       cmp     ah, 1
4567 00007B22 7210                     <1>       jb     short loc_gfvn_dir_load_err
4568 00007B24 3C41                     <1>       cmp    al, 'A'
4569 00007B26 720C                     <1>       jb      short loc_gfvn_dir_load_err
4570 00007B28 80FC02                   <1>       cmp    ah, 2
4571 00007B2B 7708                     <1>       ja      short get_FAT32_root_cluster
4572                                    <1>
4573 00007B2D E8634E0000               <1>       call    load_FAT_root_directory
4574 00007B32 730B                     <1>       jnc     short loc_get_volume_name
4575                                    <1>
4576                                    <1> loc_gfvn_dir_load_err:
4577 00007B34 C3                       <1>       retn
4578                                    <1>
4579                                    <1> get_FAT32_root_cluster:
4580 00007B35 8B4632                   <1>       mov    eax, [esi+LD_BPB+BPB_RootClus]
4581 00007B38 E8E34E0000               <1>       call    load_FAT_sub_directory
4582 00007B3D 7224                     <1>       jc     short loc_get_volume_name_retn
4583                                    <1>
4584                                    <1> loc_get_volume_name:
4585 00007B3F BE00000800               <1>         mov     esi, Directory_Buffer
4586 00007B44 6631C9                   <1>       xor    cx, cx ; 0
4587                                    <1> check_root_volume_name:
4588 00007B47 8A06                     <1>       mov    al, [esi]
4589 00007B49 08C0                     <1>       or      al, al
4590 00007B4B 7416                     <1>       jz      short loc_get_volume_name_retn
4591 00007B4D 807E0B08                 <1>       cmp     byte [esi+0Bh], 08h
4592 00007B51 7410                     <1>       je      short loc_get_volume_name_retn
4593 00007B53 663B0D[97880100]         <1>       cmp     cx, [DirBuff_LastEntry]
4594 00007B5A 7308                     <1>       jnb     short pass_check_root_volume_name
4595 00007B5C 6641                     <1>       inc     cx
4596 00007B5E 83C620                   <1>       add     esi, 32
4597 00007B61 EBE4                     <1>       jmp     short check_root_volume_name
4598                                    <1>
4599                                    <1> loc_get_volume_name_retn:
4600 00007B63 C3                       <1>       retn
```

```
4601                                    <1>
4602                                    <1> pass_check_root_volume_name:
4603 00007B64 803D[93880100]03          <1>         cmp    byte [DirBuff_FATType], 3
4604 00007B6B 7230                      <1>         jb     short loc_get_volume_name_retn_xor
4605                                    <1>
4606 00007B6D BB001C0900                <1>         mov    ebx, FAT_Buffer
4607 00007B72 BE00010900                <1>         mov    esi, Logical_DOSDisks
4608 00007B77 31C0                      <1>         xor    eax, eax
4609 00007B79 8A25[92880100]            <1>         mov    ah, [DirBuff_DRV]
4610 00007B7F 80EC41                    <1>         sub    ah, 'A'
4611 00007B82 01C6                      <1>         add    esi, eax
4612 00007B84 A1[99880100]              <1>         mov    eax, [DirBuff_Cluster]
4613 00007B89 E8AC4C0000                <1>         call   get_next_cluster
4614 00007B8E 7305                      <1>         jnc    short loc_gfvn_load_FAT32_dir_cluster
4615                                    <1>
4616 00007B90 83F801                    <1>         cmp     eax, 1
4617 00007B93 F5                        <1>         cmc
4618 00007B94 C3                        <1>         retn
4619                                    <1>
4620                                    <1> loc_gfvn_load_FAT32_dir_cluster:
4621 00007B95 E8864E0000                <1>         call   load_FAT_sub_directory
4622 00007B9A 73A3                      <1>         jnc    short loc_get_volume_name
4623 00007B9C C3                        <1>         retn
4624                                    <1>
4625                                    <1> loc_get_volume_name_retn_xor:
4626 00007B9D 31C0                      <1>         xor    eax, eax
4627 00007B9F C3                        <1>         retn
4628                                    <1>
4629                                    <1> get_media_change_status:
4630                                    <1>         ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
4631                                    <1>         ; 09/09/2009
4632                                    <1>         ; INPUT:
4633                                    <1>         ;     DL = Drive number (physical)
4634                                    <1>         ; OUTPUT: clc & AH = 6 media changed
4635                                    <1>         ;     clc & AH = 0 media not changed
4636                                    <1>         ;     stc -> Drive not ready or an error
4637                                    <1>
4638 00007BA0 B416                      <1>         mov    ah, 16h
4639 00007BA2 E8BCD6FFFF                <1>         call   int13h
4640 00007BA7 80FC06                    <1>         cmp    ah, 06h
4641 00007BAA 7405                      <1>         je     short loc_gmc_status_retn
4642 00007BAC 08E4                      <1>         or     ah, ah
4643 00007BAE 7401                      <1>         jz     short loc_gmc_status_retn
4644                                    <1> loc_gmc_status_stc_retn:
4645 00007BB0 F9                        <1>         stc
4646                                    <1> loc_gmc_status_retn:
4647 00007BB1 C3                        <1>         retn
3108                                        %include 'trdosk3.s' ; 06/01/2016
3109                                    <1> ; ****************************************************************************
3110                                    <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - MAIN PROGRAM : trdosk3.s
3111                                    <1> ; ----------------------------------------------------------------------------
3112                                    <1> ; Last Update: 31/12/2017
3113                                    <1> ; ----------------------------------------------------------------------------
3114                                    <1> ; Beginning: 06/01/2016
3115                                    <1> ; ----------------------------------------------------------------------------
3116                                    <1> ; Assembler: NASM version 2.11 (trdos386.s)
3117                                    <1> ; ----------------------------------------------------------------------------
3118                                    <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
3119                                    <1> ; MAINPROG.ASM (09/11/2011)
3120                                    <1> ; ****************************************************************************
3121                                    <1> ; MAINPROG.ASM [ TRDOS KERNEL - COMMAND EXECUTER SECTION - MAIN PROGRAM ]
3122                                    <1> ; (c) 2004-2011  Erdogan TAN  [ 17/01/2004 ]  Last Update: 09/11/2011
3123                                    <1> ; CMD_INTR.ASM [ TRDOS Command Interpreter Procedure ] Last Update: 09/11/2011
3124                                    <1> ; DIR.ASM [ DIRECTORY FUNCTIONS ] Last Update: 09/10/2011
3125                                    <1> ; FILE.ASM [ FILE FUNCTIONS ] Last Update: 09/10/2011
3126                                    <1>
3127                                    <1> change_current_drive:
3128                                    <1>         ; 16/10/2016
3129                                    <1>         ; 02/02/2016
3130                                    <1>         ; 15/01/2016 (TRDOS 386 = TRDOS v2.0)
3131                                    <1>         ; 18/08/2011
3132                                    <1>         ; 09/09/2009
3133                                    <1>         ; INPUT:
3134                                    <1>         ;   DL = Logical DOS Drive Number
3135                                    <1>         ; OUTPUT:
3136                                    <1>         ;  cf=1 -> Not successful
3137                                    <1>         ;   EAX = Error code
3138                                    <1>         ;  cf=0 ->
3139                                    <1>         ;   EAX = 0 (successful)
3140                                    <1>
3141 00007BB2 31DB                      <1>         xor    ebx, ebx
3142 00007BB4 88D7                      <1>         mov    bh, dl
3143                                    <1>
3144                                    <1>         ;cmp   dl, 1
3145                                    <1>         ;jna   short loc_ccdrv_initial_media_change_check
3146                                    <1>         ;cmp   bh, [Last_Dos_DiskNo]
3147                                    <1>         ;ja    short loc_ccdrv_drive_not_ready_err
3148                                    <1>
3149                                    <1> loc_ccdrv_initial_media_change_check:
3150 00007BB6 BE00010900                <1>         mov    esi, Logical_DOSDisks
3151 00007BBB 01DE                      <1>         add    esi, ebx
3152                                    <1> loc_ccdrv_dos_drive_name_check:
3153 00007BBD 80FA02                    <1>         cmp    dl, 2
3154 00007BC0 720F                      <1>         jb     short loc_ccdrv_dos_drive_name_check_ok
3155                                    <1>
3156 00007BC2 8A06                      <1>         mov    al, [esi+LD_Name]
3157 00007BC4 2C41                      <1>         sub    al, 'A'
3158 00007BC6 38D0                      <1>         cmp    al, dl
3159 00007BC8 7407                      <1>         je     short loc_ccdrv_dos_drive_name_check_ok
3160                                    <1>
3161                                    <1> loc_ccdrv_drive_not_ready_err:
3162                                    <1>         ; 16/10/2016 (15h -> 15)
3163 00007BCA B80F000000                <1>         mov    eax, 15 ; Drive not ready
3164                                    <1> loc_change_current_drive_stc_retn:
3165 00007BCF F9                        <1>         stc
```

```
3166 00007BD0 C3                <1>       retn
3167                            <1>
3168                            <1> loc_ccdrv_dos_drive_name_check_ok:
3169 00007BD1 8A667E            <1>       mov    ah, [esi+LD_MediaChanged]
3170 00007BD4 80FC06            <1>       cmp    ah, 6  ; VOLUME NAME CHECK/MOVE SIGN
3171 00007BD7 7455              <1>       je     short loc_ccdrv_get_FAT_volume_name_0
3172                            <1>
3173 00007BD9 80FA01            <1>       cmp    dl, 1
3174 00007BDC 777D              <1>       ja     short loc_gmcs_init_drv_hd
3175                            <1>
3176                            <1> loc_gmcs_init_drv_fd:
3177 00007BDE 08E4              <1>       or     ah, ah
3178                            <1>       ; AH = 1 is initialization sign (invalid_fd_parameter)
3179 00007BE0 7517              <1>       jnz    short loc_ccdrv_call_fd_init
3180                            <1>
3181 00007BE2 E8B9FFFFFF        <1>       call   get_media_change_status
3182 00007BE7 72E1              <1>       jc     short loc_ccdrv_drive_not_ready_err
3183                            <1>
3184 00007BE9 20E4              <1>       and    ah, ah
3185 00007BEB 7476              <1>       jz     short loc_change_current_drv3
3186                            <1>
3187 00007BED 80F406            <1>       xor    ah, 6
3188 00007BF0 75D8              <1>       jnz    short loc_ccdrv_drive_not_ready_err
3189                            <1>
3190                            <1> loc_ccdrv_call_fd_init_check_vol_id:
3191 00007BF2 E8440A0000        <1>       call   get_volume_serial_number
3192 00007BF7 730D              <1>       jnc    short loc_ccdrv_check_vol_serial
3193                            <1>
3194                            <1> loc_ccdrv_call_fd_init:
3195 00007BF9 E872FCFFFF        <1>       call   floppy_drv_init
3196 00007BFE 731A              <1>       jnc    short loc_reset_drv_fd_current_dir
3197                            <1>
3198                            <1> loc_ccdrv_fdinit_fail_retn:
3199                            <1>       ; 16/10/2016
3200 00007C00 B80F000000        <1>       mov    eax, 15 ; Drive not ready
3201 00007C05 C3                <1>       retn
3202                            <1>
3203                            <1> loc_ccdrv_check_vol_serial:
3204 00007C06 A3[64810100]      <1>       mov    [Current_VolSerial], eax
3205                            <1>       ;mov   dl, bh
3206 00007C0B E860FCFFFF        <1>       call   floppy_drv_init
3207 00007C10 72EE              <1>       jc     short loc_ccdrv_fdinit_fail_retn
3208                            <1>
3209 00007C12 3B05[64810100]    <1>       cmp    eax, [Current_VolSerial]
3210 00007C18 7445              <1>       je     short loc_change_current_drv2
3211                            <1>
3212                            <1> loc_reset_drv_fd_current_dir:
3213 00007C1A 31C0              <1>       xor    eax, eax
3214 00007C1C 88467F            <1>         mov [esi+LD_CDirLevel], al
3215 00007C1F 89F7              <1>       mov    edi, esi
3216 00007C21 81C780000000      <1>       add    edi, LD_CurrentDirectory
3217 00007C27 B920000000        <1>       mov    ecx, 32
3218 00007C2C F3AB              <1>       rep    stosd
3219                            <1>
3220                            <1> loc_ccdrv_get_FAT_volume_name_0:
3221 00007C2E 8A4603            <1>       mov    al, [esi+LD_FATType]
3222 00007C31 08C0              <1>       or     al, al
3223 00007C33 742A              <1>       jz     short loc_change_current_drv2
3224                            <1>
3225 00007C35 56                <1>       push   esi
3226 00007C36 3C02              <1>       cmp    al, 2
3227 00007C38 7705              <1>       ja     short loc_ccdrv_get_FAT32_vol_name
3228                            <1>
3229                            <1> loc_ccdrv_get_FAT2_16_vol_name:
3230 00007C3A 83C631            <1>       add    esi, LD_BPB + VolumeLabel
3231 00007C3D EB03              <1>       jmp    short loc_ccdrv_get_FAT_volume_name_1
3232                            <1>
3233                            <1> loc_ccdrv_get_FAT32_vol_name:
3234 00007C3F 83C64D            <1>       add    esi, LD_BPB + FAT32_VolLab
3235                            <1> loc_ccdrv_get_FAT_volume_name_1:
3236 00007C42 53                <1>       push   ebx
3237 00007C43 56                <1>       push   esi
3238 00007C44 E8C3FEFFFF        <1>       call   get_FAT_volume_name
3239 00007C49 5F                <1>       pop    edi
3240 00007C4A 5B                <1>       pop    ebx
3241                            <1>       ; BL = 0
3242 00007C4B 720B              <1>       jc     short loc_change_current_drv1
3243 00007C4D 20C0              <1>       and    al, al
3244 00007C4F 7407              <1>       jz     short loc_change_current_drv1
3245                            <1>
3246                            <1> loc_ccdrv_move_FAT_volume_name:
3247 00007C51 B90B000000        <1>       mov    ecx, 11
3248 00007C56 F3A4              <1>       rep    movsb
3249                            <1>
3250                            <1> loc_change_current_drv1:
3251 00007C58 5E                <1>       pop    esi
3252 00007C59 EB04              <1>       jmp    short loc_change_current_drv2
3253                            <1>
3254                            <1> loc_gmcs_init_drv_hd:
3255 00007C5B 08E4              <1>       or     ah, ah
3256 00007C5D 7404              <1>       jz     short loc_change_current_drv3
3257                            <1>       ; BL = 0, BH = Logical DOS drive number
3258                            <1> loc_change_current_drv2:
3259 00007C5F C6467E00          <1>       mov    byte [esi+LD_MediaChanged], 0
3260                            <1> loc_change_current_drv3:
3261 00007C63 883D[6E810100]    <1>       mov    [Current_Drv], bh
3262                            <1>
3263                            <1>       ;call  restore_current_directory
3264                            <1>       ;retn
3265                            <1>
3266                            <1> restore_current_directory:
3267                            <1>       ; 11/02/2016
3268                            <1>       ; 15/01/2016 (TRDOS 386 = TRDOS v2.0)
3269                            <1>       ; 25/01/2010
3270                            <1>       ; 12/10/2009
```

```
                                <1>       ;
                                <1>       ; INPUT:
                                <1>       ;   ESI = Logical DOS Drive Description Table
                                <1>       ;
                                <1>       ; OUTPUT:
                                <1>       ;   ESI = Logical DOS Drive Description Table
                                <1>       ;   EDI = offset Current_Dir_Drv
                                <1>
00007C69 8A4603                 <1>       mov    al, [esi+LD_FATType]
00007C6C A2[6D810100]           <1>       mov    [Current_FATType], al
                                <1>
00007C71 8A26                   <1>       mov    ah, [esi+LD_Name]
00007C73 8825[6F810100]         <1>       mov    [Current_Dir_Drv], ah
                                <1>
00007C79 20C0                   <1>       and    al, al
00007C7B 741D                   <1>       jz     short loc_restore_FS_current_directory
                                <1>
                                <1> loc_restore_FAT_current_directory:
00007C7D 8A667F                 <1>       mov    ah, [esi+LD_CDirLevel]
00007C80 8825[6C810100]         <1>       mov    [Current_Dir_Level], ah
00007C86 08E4                   <1>       or     ah, ah
00007C88 7416                   <1>         jz   short loc_ccdrv_reset_cdir_FAT_12_16_32_fcluster
                                <1>
00007C8A 0FB6D4                 <1>       movzx  edx, ah
00007C8D C0E204                 <1>       shl    dl, 4 ; * 16
00007C90 01F2                   <1>         add  edx, esi
00007C92 8B828C000000           <1>       mov    eax, [edx+LD_CurrentDirectory+12]
00007C98 EB2C                   <1>       jmp    short loc_ccdrv_reset_cdir_FAT_fcluster
                                <1>
                                <1> loc_restore_FS_current_directory:
00007C9A E8BC4D0000             <1>       call   load_current_FS_directory
00007C9F C3                     <1>       retn
                                <1>
                                <1> loc_ccdrv_reset_cdir_FAT_12_16_32_fcluster:
00007CA0 3C03                   <1>       cmp    al, 3
00007CA2 7205                   <1>       jb     short loc_ccdrv_reset_cdir_FAT_12_16_fcluster
                                <1> loc_ccdrv_reset_cdir_FAT32_fcluster:
00007CA4 8B4632                 <1>       mov    eax, [esi+LD_BPB+FAT32_RootFClust]
00007CA7 EB04                   <1>       jmp    short loc_ccdrv_check_rootdir_sign
                                <1> loc_ccdrv_reset_cdir_FAT_12_16_fcluster:
00007CA9 30C0                   <1>       xor    al, al  ; xor eax, eax
00007CAB 31D2                   <1>       xor    edx, edx
                                <1> loc_ccdrv_check_rootdir_sign:
00007CAD 80BE8000000000         <1>       cmp    byte [esi+LD_CurrentDirectory], 0
00007CB4 7510                   <1>       jne    short loc_ccdrv_reset_cdir_FAT_fcluster
                                <1> loc_ccdrv_set_rootdir_FAT_fcluster:
00007CB6 89868C000000           <1>         mov     [esi+LD_CurrentDirectory+12], eax
00007CBC C78680000000524F4F-    <1>       mov    dword [esi+LD_CurrentDirectory], 'ROOT'
00007CC5 54                     <1>
                                <1>
                                <1> loc_ccdrv_reset_cdir_FAT_fcluster:
00007CC6 A3[68810100]           <1>       mov    [Current_Dir_FCluster], eax
                                <1>
00007CCB BF[CB880100]           <1>       mov    edi, PATH_Array
00007CD0 89F2                   <1>       mov    edx, esi
00007CD2 81C680000000           <1>       add    esi, LD_CurrentDirectory
00007CD8 B920000000             <1>       mov    ecx, 32
00007CDD F3A5                   <1>       rep    movsd
                                <1>
00007CDF E84C2D0000             <1>       call   change_prompt_dir_string
                                <1>
00007CE4 89D6                   <1>       mov    esi, edx
                                <1>
00007CE6 29C0                   <1>         sub  eax, eax
                                <1>        ;sub  edx, edx
00007CE8 BF[6F810100]           <1>       mov    edi, Current_Dir_Drv
                                <1>
00007CED A2[55390100]           <1>       mov    [Restore_CDIR], al ; 0
00007CF2 C3                     <1>       retn
                                <1>
                                <1> dos_prompt:
                                <1>       ; 06/05/2016
                                <1>       ; 30/01/2016
                                <1>       ; 29/01/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 15/09/2011
                                <1>       ; 13/09/2009
                                <1>       ; 2004-2005
                                <1>
                                <1>       ; 06/05/2016
00007CF3 C705[288D0100]-        <1>       mov    dword [mainprog_return_addr], return_from_cmd_interpreter
00007CF9 [A77D0000]             <1>
                                <1>
                                <1> loc_TRDOS_prompt:
00007CFD BF[6E820100]           <1>       mov    edi, TextBuffer
00007D02 C6075B                 <1>       mov    byte [edi], "["
00007D05 47                     <1>       inc    edi
00007D06 BE[A8390100]           <1>       mov    esi, TRDOSPromptLabel
                                <1> get_next_prompt_label_char:
00007D0B 803E20                 <1>       cmp    byte [esi], 20h
00007D0E 7203                   <1>       jb     short pass_prompt_label
00007D10 A4                     <1>       movsb
00007D11 EBF8                   <1>       jmp    short get_next_prompt_label_char
                                <1> pass_prompt_label:
00007D13 C6075D                 <1>       mov    byte [edi], "]"
00007D16 47                     <1>       inc    edi
00007D17 C60720                 <1>       mov    byte [edi], 20h
00007D1A 47                     <1>       inc    edi
00007D1B BE[6F810100]           <1>       mov    esi, Current_Dir_Drv
00007D20 66A5                   <1>       movsw
00007D22 A4                     <1>       movsb
                                <1> loc_prompt_current_directory:
00007D23 803E20                 <1>       cmp    byte [esi], 20h
00007D26 7203                   <1>       jb     short pass_prompt_current_directory
00007D28 A4                     <1>       movsb
```

```
3374 00007D29 EBF8                <1>       jmp    short loc_prompt_current_directory
3375                              <1> pass_prompt_current_directory:
3376 00007D2B C6073E              <1>       mov    byte [edi], '>'
3377 00007D2E 47                  <1>       inc    edi
3378 00007D2F C60700              <1>       mov    byte [edi], 0
3379 00007D32 BE[6E820100]        <1>       mov    esi, TextBuffer
3380 00007D37 E809F3FFFF          <1>       call   print_msg
3381                              <1>
3382                              <1>       ;sub   bh, bh ; video page = 0
3383                              <1>       ;call  get_cpos ; get cursor position
3384 00007D3C 668B15[C6800100]    <1>       mov    dx, [CURSOR_POSN] ; video page 0
3385 00007D43 8815[CE810100]      <1>       mov    [CursorColumn], dl
3386                              <1>
3387                              <1>       ; 30/01/2016 (to show cursor on the row, again)
3388                              <1>       ; (Initial color attributes of video page 0 is 0)
3389                              <1>       ; (see: 'StartPMP' in trdos386.s)
3390                              <1>       ;
3391                              <1>       ;mov   edi, 0B8000h ; start of video page 0
3392                              <1>       ;movzx ecx, dl ; column
3393                              <1>       ;mov   al, 80
3394                              <1>       ;mul   dh
3395                              <1>       ;add   ax, cx
3396                              <1>       ;shl   ax, 1 ; character + attribute
3397                              <1>       ;add   di, ax ; (2*80*row) + (2*column)
3398                              <1>       ;neg   cl
3399                              <1>       ;add   cl, 80
3400                              <1>       ;mov   ax, 700h ;  ah = 7 (color attribute)
3401                              <1>       ;rep   stosw
3402                              <1>
3403                              <1> loc_rw_char:
3404 00007D49 E899000000          <1>       call   rw_char
3405                              <1> loc_move_command:
3406 00007D4E BE[1E820100]        <1>       mov    esi, CommandBuffer
3407 00007D53 89F7                <1>       mov    edi, esi
3408 00007D55 31C9                <1>       xor    ecx, ecx
3409                              <1> first_command_char:
3410 00007D57 AC                  <1>       lodsb
3411 00007D58 3C20                <1>       cmp    al, 20h
3412 00007D5A 772E                <1>       ja     short pass_space_control
3413 00007D5C 7241                <1>       jb     short loc_move_cmd_arguments_ok
3414 00007D5E 81FE[6D820100]      <1>       cmp    esi, CommandBuffer + 79
3415 00007D64 72F1                <1>       jb     short first_command_char
3416 00007D66 EB37                <1>       jmp    short loc_move_cmd_arguments_ok
3417                              <1>
3418                              <1> next_command_char:
3419 00007D68 AC                  <1>       lodsb
3420 00007D69 3C20                <1>       cmp    al, 20h
3421 00007D6B 771D                <1>       ja     short pass_space_control
3422 00007D6D 7230                <1>       jb     short loc_move_cmd_arguments_ok
3423                              <1>
3424                              <1> loc_1st_cmd_arg: ; 30/01/2016
3425 00007D6F AC                  <1>       lodsb
3426 00007D70 3C20                <1>       cmp    al, 20h
3427 00007D72 74FB                <1>       je     short loc_1st_cmd_arg
3428 00007D74 7229                <1>       jb     short loc_move_cmd_arguments_ok
3429                              <1>
3430 00007D76 C60700              <1>         mov     byte [edi], 0
3431 00007D79 47                  <1>       inc    edi
3432                              <1>
3433                              <1> loc_move_cmd_arguments:
3434 00007D7A AA                  <1>       stosb
3435 00007D7B 81FE[6D820100]      <1>       cmp    esi, CommandBuffer + 79
3436 00007D81 731C                <1>       jnb    short loc_move_cmd_arguments_ok
3437 00007D83 AC                  <1>         lodsb
3438 00007D84 3C20                <1>       cmp    al, 20h
3439 00007D86 73F2                <1>       jnb    short loc_move_cmd_arguments
3440 00007D88 EB15                <1>       jmp    short loc_move_cmd_arguments_ok
3441                              <1>
3442                              <1> pass_space_control:
3443 00007D8A 3C61                <1>       cmp    al, 61h
3444 00007D8C 7206                <1>       jb     short pass_capitalize
3445 00007D8E 3C7A                <1>       cmp    al, 7Ah
3446 00007D90 7702                <1>       ja     short pass_capitalize
3447 00007D92 24DF                <1>       and    al, 0DFh
3448                              <1> pass_capitalize:
3449 00007D94 AA                  <1>       stosb
3450 00007D95 FEC1                <1>       inc     cl
3451 00007D97 81FE[6D820100]      <1>         cmp     esi, CommandBuffer + 79
3452 00007D9D 72C9                <1>       jb      short next_command_char
3453                              <1>
3454                              <1> loc_move_cmd_arguments_ok:
3455 00007D9F C60700              <1>         mov     byte [edi], 0
3456                              <1>
3457                              <1> call_command_interpreter:
3458 00007DA2 E8CF080000          <1>       call    command_interpreter
3459                              <1>
3460                              <1> return_from_cmd_interpreter:
3461 00007DA7 B950000000          <1>         mov ecx, 80
3462                              <1>       ;mov   cx, 80
3463 00007DAC BF[1E820100]        <1>       mov    edi, CommandBuffer
3464 00007DB1 30C0                <1>       xor    al, al
3465 00007DB3 F3AA                <1>       rep    stosb
3466                              <1>       ;cmp   byte [Program_Exit], 0
3467                              <1>       ;ja    short loc_terminate_trdos
3468                              <1>
3469                              <1>       ; 16/01/2016
3470 00007DB5 803D[BA6A0000]03    <1>       cmp    byte [CRT_MODE], 3 ; 80*25 color
3471 00007DBC 741D                <1>       je     short pass_set_txt_mode
3472                              <1>
3473 00007DBE E8359DFFFF          <1>       call   set_txt_mode ; set vide mode to 03h
3474                              <1>       ; 07/01/2017
3475 00007DC3 30C0                <1>       xor    al, al
3476                              <1>
3477                              <1> loc_check_active_page:
3478                              <1>       ;xor   al, al
```

```
3479 00007DC5 3805[D6800100]      <1>       cmp    [ACTIVE_PAGE], al ; 0
3480 00007DCB 0F842CFFFFFF        <1>         je      loc_TRDOS_prompt
3481                              <1>       ; AL = 0 = video page 0
3482 00007DD1 E855A1FFFF          <1>       call   set_active_page
3483 00007DD6 E922FFFFFF          <1>         jmp     loc_TRDOS_prompt ; infinitive loop
3484                              <1>
3485                              <1> pass_set_txt_mode:
3486 00007DDB BE[D7430100]        <1>       mov    esi, nextline
3487 00007DE0 E860F2FFFF          <1>       call   print_msg
3488 00007DE5 EBDE                <1>       jmp     short loc_check_active_page
3489                              <1>
3490                              <1> rw_char:
3491                              <1>       ; 13/05/2016
3492                              <1>       ; 30/01/2016
3493                              <1>       ; 29/01/2016
3494                              <1>       ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
3495                              <1>       ; 2004-2005
3496                              <1>
3497                              <1>       ; DH = cursor row, DL = cursor column
3498                              <1>       ; BH = 0 = video page number (active page)
3499                              <1>
3500                              <1>       ;xor   bh, bh ; 0 = video page 0
3501                              <1>
3502                              <1> readnextchar:
3503 00007DE7 30E4                <1>       xor     ah, ah
3504 00007DE9 E8ED90FFFF          <1>       call   int16h
3505 00007DEE 20C0                <1>       and    al, al
3506 00007DF0 7432                <1>       jz     short loc_arrow
3507 00007DF2 3CE0                <1>       cmp    al, 0E0h
3508 00007DF4 742E                <1>       je     short loc_arrow
3509 00007DF6 3C08                <1>       cmp    al, 08h
3510 00007DF8 7542                <1>       jne    short char_return
3511                              <1> loc_back:
3512 00007DFA 3A15[CE810100]      <1>       cmp    dl, [CursorColumn]
3513 00007E00 76E5                <1>       jna     short readnextchar
3514                              <1> prev_column:
3515 00007E02 FECA                <1>       dec    dl
3516                              <1> set_cursor_pos:
3517                              <1>       ;push  dx
3518 00007E04 52                  <1>       push   edx ; 29/12/2017
3519                              <1>       ;xor   bh, bh ; 0 = video page 0
3520                              <1>       ; DH = Row, DL = Column
3521 00007E05 E8F0A4FFFF          <1>       call   _set_cpos ; 17/01/2016
3522 00007E0A 5A                  <1>         pop  edx ; 29/12/2017
3523                              <1>       ;pop   dx
3524                              <1>       ;movzx ebx, dl
3525 00007E0B 88D3                <1>       mov    bl, dl
3526 00007E0D 2A1D[CE810100]      <1>       sub    bl, [CursorColumn]
3527 00007E13 B020                <1>       mov    al, 20h
3528 00007E15 8883[1E820100]      <1>       mov    [CommandBuffer+ebx], al
3529                              <1>       ;sub   bh, bh ; video page 0
3530                              <1>       ;mov   cx, 1
3531 00007E1B B307                <1>       mov    bl, 7 ; color attribute
3532 00007E1D E8C0A3FFFF          <1>       call   _write_c_current ; 17/01/2016
3533                              <1>       ;mov   dx, [CURSOR_POSN]
3534 00007E22 EBC3                <1>       jmp    short readnextchar
3535                              <1> loc_arrow:
3536 00007E24 80FC4B              <1>       cmp    ah, 4Bh
3537 00007E27 74D1                <1>       je     short loc_back
3538 00007E29 80FC53              <1>       cmp    ah, 53h
3539 00007E2C 74CC                <1>       je      short loc_back
3540 00007E2E 80FC4D              <1>       cmp    ah, 4Dh
3541 00007E31 75B4                <1>       jne    short readnextchar
3542 00007E33 80FA4F              <1>       cmp    dl, 79
3543 00007E36 73AF                <1>       jnb    short readnextchar
3544 00007E38 FEC2                <1>       inc    dl
3545 00007E3A EBC8                <1>       jmp    short set_cursor_pos
3546                              <1> char_return:
3547 00007E3C 0FB6DA              <1>       movzx  ebx, dl
3548 00007E3F 2A1D[CE810100]      <1>       sub    bl, [CursorColumn]
3549 00007E45 3C20                <1>       cmp    al, 20h
3550 00007E47 721D                <1>       jb     short loc_escape
3551 00007E49 8883[1E820100]      <1>       mov    [CommandBuffer+ebx], al
3552 00007E4F 80FA4F              <1>       cmp    dl, 79
3553 00007E52 7393                <1>       jnb    short readnextchar
3554 00007E54 66BB0700            <1>       mov    bx, 7 ; color attribute
3555 00007E58 E806A4FFFF          <1>       call   _write_tty
3556 00007E5D 668B15[C6800100]    <1>       mov    dx, [CURSOR_POSN] ; video page 0
3557 00007E64 EB81                <1>         jmp     readnextchar
3558                              <1> loc_escape:
3559 00007E66 3C1B                <1>       cmp    al, 1Bh
3560 00007E68 7418                <1>       je     short rw_char_retn
3561                              <1>       ;
3562 00007E6A 3C0D                <1>       cmp    al, 0Dh ; CR
3563 00007E6C 0F8575FFFFFF        <1>         jne     readnextchar
3564                              <1>       ; 13/05/2016
3565 00007E72 66BB0700            <1>       mov    bx, 7 ; attribute/color (bl)
3566                              <1>                    ; video page 0 (bh=0)
3567 00007E76 E8E8A3FFFF          <1>       call   _write_tty
3568                              <1>       ;mov   bx, 7  ; attribute/color
3569                              <1>                    ; video page 0 (bh=0)
3570 00007E7B B00A                <1>       mov    al, 0Ah ; LF
3571 00007E7D E8E1A3FFFF          <1>       call   _write_tty
3572                              <1> rw_char_retn:
3573 00007E82 C3                  <1>       retn
3574                              <1>
3575                              <1> show_date:
3576                              <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
3577                              <1>         ; 2004-2005
3578                              <1>
3579                              <1>       ;mov   ah, 04h
3580                              <1>       ;call  int1Ah
3581 00007E83 E890E7FFFF          <1>       call   RTC_40 ; GET RTC DATE
3582                              <1>
3583 00007E88 88D0                <1>       mov    al, dl
```

```
3584 00007E8A E83F90FFFF          <1>       call   bcd_to_ascii
3585 00007E8F 66A3[943A0100]      <1>       mov    [Day], ax
3586                              <1>
3587 00007E95 88F0                <1>       mov    al, dh
3588 00007E97 E83290FFFF          <1>       call   bcd_to_ascii
3589 00007E9C 66A3[973A0100]      <1>       mov    [Month], ax
3590                              <1>
3591 00007EA2 88E8                <1>       mov    al, ch
3592 00007EA4 E82590FFFF          <1>       call   bcd_to_ascii
3593 00007EA9 66A3[9A3A0100]      <1>       mov    [Century], ax
3594                              <1>
3595 00007EAF 88C8                <1>       mov    al, cl
3596 00007EB1 E81890FFFF          <1>       call   bcd_to_ascii
3597 00007EB6 66A3[9C3A0100]      <1>       mov    word [Year], ax
3598                              <1>
3599 00007EBC BE[843A0100]        <1>       mov    esi, Msg_Show_Date
3600 00007EC1 E87FF1FFFF          <1>       call   print_msg
3601                              <1>
3602 00007EC6 C3                  <1>       retn
3603                              <1>
3604                              <1> set_date:
3605                              <1>       ; 13/05/2016
3606                              <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
3607                              <1>         ; 2004-2005
3608                              <1>
3609 00007EC7 BE[683A0100]        <1>       mov    esi, Msg_Enter_Date
3610 00007ECC E874F1FFFF          <1>       call   print_msg
3611                              <1>
3612                              <1> loc_enter_day_1:
3613 00007ED1 30E4                <1>       xor     ah, ah
3614 00007ED3 E80390FFFF          <1>       call   int16h
3615                              <1>       ; AL = ASCII Code of the Character
3616 00007ED8 3C0D                <1>       cmp    al, 13
3617 00007EDA 0F84B7010000        <1>       je     loc_set_date_retn
3618 00007EE0 3C1B                <1>       cmp    al, 27
3619 00007EE2 0F84AF010000        <1>       je     loc_set_date_retn
3620 00007EE8 A2[943A0100]        <1>       mov    [Day], al
3621 00007EED 3C30                <1>       cmp    al, '0'
3622 00007EEF 0F82AD010000        <1>       jb     loc_set_date_stc_0
3623 00007EF5 3C33                <1>       cmp    al, '3'
3624 00007EF7 0F87A5010000        <1>       ja     loc_set_date_stc_0
3625                              <1>       ; 13/05/2016
3626                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3627                              <1>                    ; video page 0 (bh)
3628 00007EFD B307                <1>       mov    bl, 7
3629 00007EFF E85FA3FFFF          <1>       call   _write_tty
3630                              <1> loc_enter_day_2:
3631 00007F04 30E4                <1>       xor     ah, ah
3632 00007F06 E8D08FFFFF          <1>       call   int16h
3633                              <1>       ; AL = ASCII Code of the Character
3634 00007F0B 3C1B                <1>       cmp    al, 27
3635 00007F0D 0F8484010000        <1>         je      loc_set_date_retn
3636 00007F13 A2[953A0100]        <1>       mov    [Day+1], al
3637 00007F18 3C30                <1>       cmp    al, '0'
3638 00007F1A 0F828C010000        <1>         jb      loc_set_date_stc_1
3639 00007F20 3C39                <1>       cmp    al, '9'
3640 00007F22 0F8784010000        <1>         ja      loc_set_date_stc_1
3641 00007F28 803D[943A0100]33    <1>       cmp    byte [Day], '3'
3642 00007F2F 7208                <1>       jb     short pass_set_day_31
3643 00007F31 3C31                <1>       cmp    al, '1'
3644 00007F33 0F8773010000        <1>         ja      loc_set_date_stc_1
3645                              <1> pass_set_day_31:
3646                              <1>       ; 13/05/2016
3647                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3648                              <1>                    ; video page 0 (bh)
3649 00007F39 B307                <1>       mov    bl, 7
3650 00007F3B E823A3FFFF          <1>       call   _write_tty
3651                              <1> loc_enter_separator_1:
3652 00007F40 28E4                <1>       sub     ah, ah ; 0
3653 00007F42 E8948FFFFF          <1>       call   int16h
3654                              <1>       ; AL = ASCII Code of the Character
3655 00007F47 3C1B                <1>       cmp    al, 27
3656 00007F49 0F8448010000        <1>         je      loc_set_date_retn
3657 00007F4F 3C2D                <1>       cmp    al, '-'
3658 00007F51 7408                <1>       je     short pass_set_date_separator_1
3659 00007F53 3C2F                <1>       cmp    al, '/'
3660 00007F55 0F856C010000        <1>         jne     loc_set_date_stc_2
3661                              <1> pass_set_date_separator_1:
3662                              <1>       ; 13/05/2016
3663                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3664                              <1>                    ; video page 0 (bh)
3665 00007F5B B307                <1>       mov    bl, 7
3666 00007F5D E801A3FFFF          <1>       call   _write_tty
3667                              <1> loc_enter_month_1:
3668 00007F62 30E4                <1>       xor     ah, ah ; 0
3669 00007F64 E8728FFFFF          <1>       call   int16h
3670                              <1>       ; AL = ASCII Code of the Character
3671 00007F69 3C1B                <1>       cmp    al, 27
3672 00007F6B 0F8426010000        <1>         je      loc_set_date_retn
3673 00007F71 A2[973A0100]        <1>       mov    [Month], al
3674 00007F76 3C30                <1>       cmp    al, '0'
3675 00007F78 0F8264010000        <1>         jb      loc_set_date_stc_3
3676 00007F7E 3C31                <1>       cmp    al, '1'
3677 00007F80 0F875C010000        <1>         ja      loc_set_date_stc_3
3678                              <1>       ; 13/05/2016
3679                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3680                              <1>                      ; video page 0 (bh)
3681 00007F86 B307                <1>       mov    bl, 7
3682 00007F88 E8D6A2FFFF          <1>       call   _write_tty
3683                              <1> loc_enter_month_2:
3684 00007F8D 30E4                <1>       xor     ah, ah
3685 00007F8F E8478FFFFF          <1>       call   int16h
3686                              <1>       ; AL = ASCII Code of the Character
3687 00007F94 3C1B                <1>       cmp    al, 27
3688 00007F96 0F84FB000000        <1>         je      loc_set_date_retn
```

```
3689 00007F9C A2[983A0100]        <1>       mov    [Month+1], al
3690 00007FA1 3C30                <1>       cmp    al, '0'
3691 00007FA3 0F8254010000        <1>         jb      loc_set_date_stc_4
3692 00007FA9 3C39                <1>       cmp    al, '9'
3693 00007FAB 0F874C010000        <1>         ja      loc_set_date_stc_4
3694 00007FB1 803D[973A0100]31    <1>       cmp    byte [Month], '1'
3695 00007FB8 7208                <1>       jb     short pass_set_month_12
3696 00007FBA 3C32                <1>       cmp    al, '2'
3697 00007FBC 0F873B010000        <1>         ja      loc_set_date_stc_4
3698                              <1> pass_set_month_12:
3699                              <1>       ; 13/05/2016
3700                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3701                              <1>                    ; video page 0 (bh)
3702 00007FC2 B307                <1>       mov    bl, 7
3703 00007FC4 E89AA2FFFF          <1>       call   _write_tty
3704                              <1> loc_enter_separator_2:
3705 00007FC9 28E4                <1>       sub     ah, ah
3706 00007FCB E80B8FFFFF          <1>       call   int16h
3707                              <1>       ; AL = ASCII Code of the Character
3708 00007FD0 3C1B                <1>       cmp    al, 27
3709 00007FD2 0F84BF000000        <1>         je      loc_set_date_retn
3710 00007FD8 3C2D                <1>       cmp    al, '-'
3711 00007FDA 7408                <1>       je     short pass_set_date_separator_2
3712 00007FDC 3C2F                <1>       cmp    al, '/'
3713 00007FDE 0F8534010000        <1>         jne     loc_set_date_stc_5
3714                              <1> pass_set_date_separator_2:
3715                              <1>       ; 13/05/2016
3716                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3717                              <1>                    ; video page 0 (bh)
3718 00007FE4 B307                <1>       mov    bl, 7
3719 00007FE6 E878A2FFFF          <1>       call   _write_tty
3720                              <1> loc_enter_year_1:
3721 00007FEB 30E4                <1>       xor     ah, ah
3722 00007FED E8E98EFFFF          <1>       call   int16h
3723                              <1>       ; AL = ASCII Code of the Character
3724 00007FF2 3C1B                <1>       cmp    al, 27
3725 00007FF4 0F849D000000        <1>         je      loc_set_date_retn
3726 00007FFA A2[9C3A0100]        <1>       mov    [Year], al
3727 00007FFF 3C30                <1>       cmp    al, '0'
3728 00008001 0F822C010000        <1>         jb      loc_set_date_stc_6
3729 00008007 3C39                <1>       cmp    al, '9'
3730 00008009 0F8724010000        <1>         ja      loc_set_date_stc_6
3731                              <1>       ; 13/05/2016
3732                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3733                              <1>                    ; video page 0 (bh)
3734 0000800F B307                <1>       mov    bl, 7
3735 00008011 E84DA2FFFF          <1>       call   _write_tty
3736                              <1> loc_enter_year_2:
3737 00008016 30E4                <1>       xor    ah, ah
3738 00008018 E8BE8EFFFF          <1>       call   int16h
3739                              <1>       ; AL = ASCII Code of the Character
3740 0000801D 3C1B                <1>       cmp    al, 27
3741 0000801F 7476                <1>       je     short loc_set_date_retn
3742 00008021 A2[9D3A0100]        <1>       mov    byte [Year+1], al
3743 00008026 3C30                <1>       cmp    al, '0'
3744 00008028 0F8220010000        <1>         jb      loc_set_date_stc_7
3745 0000802E 3C39                <1>       cmp    al, '9'
3746 00008030 0F8718010000        <1>         ja      loc_set_date_stc_7
3747                              <1>       ; 13/05/2016
3748                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3749                              <1>                    ; video page 0 (bh)
3750 00008036 B307                <1>       mov    bl, 7
3751 00008038 E826A2FFFF          <1>       call   _write_tty
3752                              <1> loc_set_date_get_lchar_again:
3753 0000803D 28E4                <1>       sub    ah, ah ; 0
3754 0000803F E8978EFFFF          <1>       call   int16h
3755                              <1>       ; AL = ASCII Code of the Character
3756 00008044 3C0D                <1>       cmp    al, 13 ; ENTER key
3757 00008046 7412                <1>       je     short loc_set_date_progress
3758 00008048 3C1B                <1>       cmp    al, 27 ; ESC key
3759 0000804A 744B                <1>       je     short loc_set_date_retn
3760                              <1>       ;
3761 0000804C E82A010000          <1>       call   check_for_backspace
3762 00008051 75EA                <1>       jne    short loc_set_date_get_lchar_again
3763                              <1>
3764                              <1> loc_set_date_bs_8:
3765 00008053 E811010000          <1>       call   write_backspace
3766 00008058 EBBC                <1>       jmp    short loc_enter_year_2
3767                              <1>
3768                              <1> loc_set_date_progress:
3769                              <1>       ; Get Current Date
3770                              <1>       ;mov   ah, 04h
3771                              <1>       ;call  int1Ah
3772 0000805A E8B9E5FFFF          <1>       call   RTC_40 ; GET RTC DATE
3773                              <1>       ; CH = century (in BCD)
3774                              <1>
3775 0000805F 66A1[9C3A0100]      <1>       mov    ax, [Year]
3776 00008065 662D3030            <1>       sub    ax, '00'
3777 00008069 C0E004              <1>       shl    al, 4 ; * 16
3778 0000806C 88C1                <1>       mov    cl, al
3779 0000806E 00E1                <1>       add    cl, ah
3780 00008070 66A1[973A0100]      <1>       mov    ax, [Month]
3781 00008076 662D3030            <1>       sub    ax, '00'
3782 0000807A C0E004              <1>       shl    al, 4 ; * 16
3783 0000807D 88C6                <1>       mov    dh, al
3784 0000807F 00E6                <1>       add    dh, ah
3785 00008081 66A1[943A0100]      <1>       mov    ax, [Day]
3786 00008087 662D3030            <1>       sub    ax, '00'
3787 0000808B C0E004              <1>       shl    al, 4 ; * 16
3788 0000808E 88C2                <1>       mov    dl, al
3789 00008090 00E2                <1>       add    dl, ah
3790                              <1>
3791                              <1>       ;mov   ah, 05h
3792                              <1>       ;call  int1Ah
3793 00008092 E8AEE5FFFF          <1>       call   RTC_50 ; SET RTC DATE
```

```
                                <1>
                                <1> loc_set_date_retn:
00008097 BE[D7430100]           <1>       mov   esi, nextline
0000809C E8A4EFFFFF             <1>       call  print_msg
000080A1 C3                     <1>       retn
                                <1>
                                <1> loc_set_date_stc_0:
                                <1>       ;xor  bh, bh ; video page 0
000080A2 E8A5A2FFFF             <1>       call  beeper ; BEEP !
000080A7 E925FEFFFF             <1>         jmp     loc_enter_day_1
                                <1> loc_set_date_stc_1:
000080AC E8CA000000             <1>       call  check_for_backspace
000080B1 740A                   <1>       je    short loc_set_date_bs_1
                                <1>       ;xor  bh, bh ; video page 0
000080B3 E894A2FFFF             <1>       call  beeper ; BEEP !
000080B8 E947FEFFFF             <1>         jmp     loc_enter_day_2
                                <1> loc_set_date_bs_1:
000080BD E8A7000000             <1>       call  write_backspace
000080C2 E90AFEFFFF             <1>         jmp     loc_enter_day_1
                                <1> loc_set_date_stc_2:
000080C7 E8AF000000             <1>       call  check_for_backspace
000080CC 740A                   <1>       je    short loc_set_date_bs_2
                                <1>       ;xor  bh, bh ; video page 0
000080CE E879A2FFFF             <1>       call  beeper ; BEEP !
000080D3 E968FEFFFF             <1>         jmp     loc_enter_separator_1
                                <1> loc_set_date_bs_2:
000080D8 E88C000000             <1>       call  write_backspace
000080DD E922FEFFFF             <1>         jmp     loc_enter_day_2
                                <1> loc_set_date_stc_3:
000080E2 E894000000             <1>       call  check_for_backspace
000080E7 740A                   <1>       je short loc_set_date_bs_3
                                <1>       ;xor  bh, bh ; video page 0
000080E9 E85EA2FFFF             <1>       call  beeper ; BEEP !
000080EE E96FFEFFFF             <1>         jmp     loc_enter_month_1
                                <1> loc_set_date_bs_3:
000080F3 E871000000             <1>       call  write_backspace
000080F8 E943FEFFFF             <1>         jmp     loc_enter_separator_1
                                <1> loc_set_date_stc_4:
000080FD E879000000             <1>       call  check_for_backspace
00008102 740A                   <1>       je    short loc_set_date_bs_4
                                <1>       ;xor  bh, bh ; video page 0
00008104 E843A2FFFF             <1>       call  beeper ; BEEP !
00008109 E97FFEFFFF             <1>         jmp     loc_enter_month_2
                                <1> loc_set_date_bs_4:
0000810E E856000000             <1>       call  write_backspace
00008113 E94AFEFFFF             <1>         jmp     loc_enter_month_1
                                <1> loc_set_date_stc_5:
00008118 E85E000000             <1>       call  check_for_backspace
0000811D 740A                   <1>       je    short loc_set_date_bs_5
                                <1>       ;xor  bh, bh ; video page 0
0000811F E828A2FFFF             <1>       call  beeper ; BEEP !
00008124 E9A0FEFFFF             <1>         jmp     loc_enter_separator_2
                                <1> loc_set_date_bs_5:
00008129 E83B000000             <1>       call  write_backspace
0000812E E95AFEFFFF             <1>         jmp     loc_enter_month_2
                                <1> loc_set_date_stc_6:
00008133 E843000000             <1>       call  check_for_backspace
00008138 740A                   <1>         je      short  loc_set_date_bs_6
                                <1>       ;xor  bh, bh ; video page 0
0000813A E80DA2FFFF             <1>       call  beeper ; BEEP !
0000813F E9A7FEFFFF             <1>         jmp     loc_enter_year_1
                                <1> loc_set_date_bs_6:
00008144 E820000000             <1>       call  write_backspace
00008149 E97BFEFFFF             <1>         jmp     loc_enter_separator_2
                                <1> loc_set_date_stc_7:
0000814E E828000000             <1>       call  check_for_backspace
00008153 740A                   <1>       je    short loc_set_date_bs_7
                                <1>       ;xor  bh, bh ; video page 0
00008155 E8F2A1FFFF             <1>       call  beeper ; BEEP !
0000815A E9B7FEFFFF             <1>         jmp     loc_enter_year_2
                                <1> loc_set_date_bs_7:
0000815F E805000000             <1>       call  write_backspace
00008164 E982FEFFFF             <1>         jmp     loc_enter_year_1
                                <1>
                                <1> write_backspace:
                                <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
00008169 B008                   <1>       mov   al, 08h ; BACKSPACE
                                <1>       ; 13/05/2016
0000816B 66BB0700               <1>       mov   bx, 7 ; bl = attribute/color
                                <1>                   ; bh = video page = 0
0000816F E8EFA0FFFF             <1>       call  _write_tty
00008174 B020                   <1>       mov   al, 20h ; BLANK/SPACE char
                                <1>       ;mov  bx, 7 ; attribute/color
                                <1>       ;call _write_c_current
                                <1>       ;retn
00008176 E967A0FFFF             <1>       jmp   _write_c_current
                                <1>
                                <1> check_for_backspace:
                                <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
0000817B 663D080E               <1>       cmp   ax, 0E08h
0000817F 7410                   <1>       je    short cfbs_retn
00008181 663DE04B               <1>       cmp   ax, 4BE0h
00008185 740A                   <1>       je    short cfbs_retn
00008187 663D004B               <1>       cmp   ax, 4B00h
0000818B 7404                   <1>       je    short cfbs_retn
0000818D 663DE053               <1>       cmp   ax, 53E0h
                                <1> cfbs_retn:
00008191 C3                     <1>       retn
                                <1>
                                <1> show_time:
                                <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>         ; 2004-2005
                                <1>
                                <1>       ;mov  ah, 02h
                                <1>       ;call int1Ah
```

```
3899 00008192 E810E4FFFF          <1>       call   RTC_20 ; GET RTC TIME
3900                              <1>
3901 00008197 88E8                <1>       mov    al, ch
3902 00008199 E8308DFFFF          <1>       call   bcd_to_ascii
3903 0000819E 66A3[C23A0100]      <1>       mov    [Hour], ax
3904                              <1>
3905 000081A4 88C8                <1>       mov    al, cl
3906 000081A6 E8238DFFFF          <1>       call   bcd_to_ascii
3907 000081AB 66A3[C53A0100]      <1>       mov    [Minute], ax
3908                              <1>
3909 000081B1 88F0                <1>       mov    al, dh
3910 000081B3 E8168DFFFF          <1>       call   bcd_to_ascii
3911 000081B8 66A3[C83A0100]      <1>       mov    [Second], ax
3912                              <1>
3913 000081BE BE[B23A0100]        <1>       mov    esi, Msg_Show_Time
3914 000081C3 E87DEEFFFF          <1>       call   print_msg
3915 000081C8 C3                  <1>       retn
3916                              <1>
3917                              <1> set_time:
3918                              <1>       ; 13/05/2016
3919                              <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
3920                              <1>         ; 2004-2005
3921                              <1>
3922 000081C9 BE[A13A0100]        <1>       mov    esi, Msg_Enter_Time
3923 000081CE E872EEFFFF          <1>       call   print_msg
3924                              <1>
3925                              <1> loc_enter_hour_1:
3926 000081D3 30E4                <1>       xor     ah, ah
3927 000081D5 E8018DFFFF          <1>       call   int16h
3928                              <1>       ; AL = ASCII Code of the Character
3929 000081DA 3C0D                <1>       cmp    al, 13 ; ENTER key
3930 000081DC 0F84AE010000        <1>         je      loc_set_time_retn
3931 000081E2 3C1B                <1>       cmp    al, 27 ; ESC key
3932 000081E4 0F84A6010000        <1>         je      loc_set_time_retn
3933 000081EA A2[C23A0100]        <1>       mov    [Hour], al
3934 000081EF 3C30                <1>       cmp    al, '0'
3935 000081F1 0F82A4010000        <1>         jb      loc_set_time_stc_0
3936 000081F7 3C32                <1>       cmp    al, '2'
3937 000081F9 0F879C010000        <1>         ja      loc_set_time_stc_0
3938                              <1>       ; 13/05/2016
3939                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3940                              <1>                    ; video page 0 (bh)
3941 000081FF B307                <1>       mov    bl, 7
3942 00008201 E85DA0FFFF          <1>       call   _write_tty
3943                              <1> loc_enter_hour_2:
3944 00008206 30E4                <1>       xor     ah, ah
3945 00008208 E8CE8CFFFF          <1>       call   int16h
3946                              <1>       ; AL = ASCII Code of the Character
3947 0000820D 3C1B                <1>       cmp    al, 27
3948 0000820F 0F847B010000        <1>         je      loc_set_time_retn
3949 00008215 A2[C33A0100]        <1>       mov    [Hour+1], al
3950 0000821A 3C30                <1>       cmp    al, '0'
3951 0000821C 0F8283010000        <1>         jb      loc_set_time_stc_1
3952 00008222 3C39                <1>       cmp    al, '9'
3953 00008224 0F877B010000        <1>       ja     loc_set_time_stc_1
3954 0000822A 803D[C23A0100]32    <1>         cmp     byte [Hour], '2'
3955 00008231 7208                <1>       jb     short pass_set_time_24
3956 00008233 3C34                <1>       cmp    al, '4'
3957 00008235 0F876A010000        <1>         ja      loc_set_time_stc_1
3958                              <1> pass_set_time_24:
3959                              <1>       ; 13/05/2016
3960                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3961                              <1>                    ; video page 0 (bh)
3962 0000823B B307                <1>       mov    bl, 7
3963 0000823D E821A0FFFF          <1>       call   _write_tty
3964                              <1> loc_enter_time_separator_1:
3965 00008242 28E4                <1>       sub    ah, ah ; 0
3966 00008244 E8928CFFFF          <1>       call   int16h
3967                              <1>       ; AL = ASCII Code of the Character
3968 00008249 3C1B                <1>       cmp    al, 27
3969 0000824B 0F843F010000        <1>         je      loc_set_time_retn
3970 00008251 3C3A                <1>       cmp    al, ':'
3971 00008253 0F8567010000        <1>         jne     loc_set_time_stc_2
3972                              <1>       ; 13/05/2016
3973                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3974                              <1>                    ; video page 0 (bh)
3975 00008259 B307                <1>       mov    bl, 7
3976 0000825B E803A0FFFF          <1>       call   _write_tty
3977                              <1> loc_enter_minute_1:
3978 00008260 30E4                <1>       xor     ah, ah
3979 00008262 E8748CFFFF          <1>       call   int16h
3980                              <1>       ; AL = ASCII Code of the Character
3981 00008267 3C1B                <1>       cmp    al, 27
3982 00008269 0F8421010000        <1>         je      loc_set_time_retn
3983 0000826F A2[C53A0100]        <1>       mov    [Minute], al
3984 00008274 3C30                <1>       cmp    al, '0'
3985 00008276 0F825F010000        <1>         jb      loc_set_time_stc_3
3986 0000827C 3C35                <1>       cmp    al, '5'
3987 0000827E 0F8757010000        <1>         ja      loc_set_time_stc_3
3988                              <1>       ; 13/05/2016
3989                              <1>       ;mov   bx, 7 ; attribute/color (bl)
3990                              <1>                    ; video page 0 (bh)
3991 00008284 B307                <1>       mov    bl, 7
3992 00008286 E8D89FFFFF          <1>       call   _write_tty
3993                              <1> loc_enter_minute_2:
3994 0000828B 30E4                <1>       xor     ah, ah
3995 0000828D E8498CFFFF          <1>       call   int16h
3996                              <1>       ; AL = ASCII Code of the Character
3997 00008292 3C1B                <1>       cmp    al, 27
3998 00008294 0F84F6000000        <1>         je      loc_set_time_retn
3999 0000829A A2[C63A0100]        <1>       mov    [Minute+1], al
4000 0000829F 3C30                <1>       cmp    al, '0'
4001 000082A1 0F824F010000        <1>         jb      loc_set_time_stc_4
4002 000082A7 3C39                <1>       cmp    al, '9'
4003 000082A9 0F8747010000        <1>         ja      loc_set_time_stc_4
```

```
4004                                <1>       ; 13/05/2016
4005                                <1>       ;mov   bx, 7 ; attribute/color (bl)
4006                                <1>                    ; video page 0 (bh)
4007 000082AF B307                  <1>       mov    bl, 7
4008 000082B1 E8AD9FFFFF            <1>       call   _write_tty
4009                                <1> loc_enter_time_separator_2:
4010 000082B6 66C705[C83A0100]30-   <1>       mov    word [Second], 3030h
4010 000082BE 30                    <1>
4011 000082BF 28E4                  <1>       sub     ah, ah
4012 000082C1 E8158CFFFF            <1>       call   int16h
4013                                <1>       ; AL = ASCII Code of the Character
4014 000082C6 3C0D                  <1>       cmp    al, 13
4015 000082C8 0F8485000000          <1>         je      loc_set_time_progress
4016 000082CE 3C1B                  <1>       cmp    al, 27
4017 000082D0 0F84BA000000          <1>         je      loc_set_time_retn
4018 000082D6 3C3A                  <1>       cmp    al, ':'
4019 000082D8 0F8533010000          <1>         jne     loc_set_time_stc_5
4020                                <1>       ; 13/05/2016
4021                                <1>       ;mov   bx, 7 ; attribute/color (bl)
4022                                <1>                    ; video page 0 (bh)
4023 000082DE B307                  <1>       mov    bl, 7
4024 000082E0 E87E9FFFFF            <1>       call   _write_tty
4025                                <1> loc_enter_second_1:
4026 000082E5 30E4                  <1>       xor     ah, ah
4027 000082E7 E8EF8BFFFF            <1>       call   int16h
4028                                <1>       ; AL = ASCII Code of the Character
4029 000082EC 3C0D                  <1>       cmp    al, 13
4030 000082EE 7463                  <1>       je     short loc_set_time_progress
4031 000082F0 3C1B                  <1>       cmp    al, 27
4032 000082F2 0F8498000000          <1>         je      loc_set_time_retn
4033 000082F8 A2[C83A0100]          <1>       mov    [Second], al
4034 000082FD 3C30                  <1>       cmp    al, '0'
4035 000082FF 0F8227010000          <1>         jb      loc_set_time_stc_6
4036 00008305 3C35                  <1>       cmp    al, '5'
4037 00008307 0F871F010000          <1>         ja      loc_set_time_stc_6
4038                                <1>       ; 13/05/2016
4039                                <1>       ;mov   bx, 7 ; attribute/color (bl)
4040                                <1>                    ; video page 0 (bh)
4041 0000830D B307                  <1>       mov    bl, 7
4042 0000830F E84F9FFFFF            <1>       call   _write_tty
4043                                <1> loc_enter_second_2:
4044 00008314 30E4                  <1>       xor     ah, ah
4045 00008316 E8C08BFFFF            <1>       call   int16h
4046                                <1>       ; AL = ASCII Code of the Character
4047 0000831B 3C1B                  <1>       cmp    al, 27
4048 0000831D 7471                  <1>       je     short loc_set_time_retn
4049 0000831F 3C30                  <1>       cmp    al, '0'
4050 00008321 0F8229010000          <1>         jb      loc_set_time_stc_7
4051 00008327 3C39                  <1>       cmp    al, '9'
4052 00008329 0F8721010000          <1>         ja      loc_set_time_stc_7
4053                                <1>       ; 13/05/2016
4054                                <1>       ;mov   bx, 7 ; attribute/color (bl)
4055                                <1>                    ; video page 0 (bh)
4056 0000832F B307                  <1>       mov    bl, 7
4057 00008331 E82D9FFFFF            <1>       call   _write_tty
4058                                <1> loc_set_time_get_lchar_again:
4059 00008336 28E4                  <1>       sub    ah, ah ; 0
4060 00008338 E89E8BFFFF            <1>       call   int16h
4061                                <1>       ; AL = ASCII Code of the Character
4062 0000833D 3C0D                  <1>       cmp    al, 13
4063 0000833F 7412                  <1>       je     short loc_set_time_progress
4064 00008341 3C1B                  <1>       cmp    al, 27
4065 00008343 744B                  <1>       je     short loc_set_time_retn
4066                                <1>       ;
4067 00008345 E831FEFFFF            <1>       call   check_for_backspace
4068 0000834A 75EA                  <1>       jne    short loc_set_time_get_lchar_again
4069                                <1>
4070                                <1> loc_set_time_bs_8:
4071 0000834C E818FEFFFF            <1>       call   write_backspace
4072 00008351 EBC1                  <1>       jmp    short loc_enter_second_2
4073                                <1>
4074                                <1> loc_set_time_progress:
4075                                <1>       ; Get Current Time
4076                                <1>       ;mov   ah, 02h
4077                                <1>       ;call  int1Ah
4078 00008353 E84FE2FFFF            <1>       call   RTC_20 ; GET RTC TIME
4079                                <1>       ;DL = Daylight Savings Enable option (0-1)
4080                                <1>
4081 00008358 66A1[C23A0100]        <1>       mov    ax, [Hour]
4082 0000835E 662D3030              <1>       sub    ax, '00'
4083 00008362 C0E004                <1>       shl    al, 4 ; * 16
4084 00008365 88C5                  <1>       mov    ch, al
4085 00008367 00E5                  <1>       add    ch, ah
4086 00008369 66A1[C53A0100]        <1>       mov    ax, [Minute]
4087 0000836F 662D3030              <1>       sub    ax, '00'
4088 00008373 C0E004                <1>       shl    al, 4 ; * 16
4089 00008376 88C1                  <1>       mov    cl, al
4090 00008378 00E1                  <1>       add    cl, ah
4091 0000837A 66A1[C83A0100]        <1>       mov    ax, [Second]
4092 00008380 662D3030              <1>       sub    ax, '00'
4093 00008384 C0E004                <1>       shl    al, 4 ; * 16
4094 00008387 88C6                  <1>       mov    dh, al
4095 00008389 00E6                  <1>       add    dh, ah
4096                                <1>
4097                                <1>       ;mov   ah, 03h
4098                                <1>       ;call  int1Ah
4099 0000838B E846E2FFFF            <1>       call   RTC_30 ; SET RTC TIME
4100                                <1>
4101                                <1> loc_set_time_retn:
4102 00008390 BE[D7430100]          <1>       mov    esi, nextline
4103 00008395 E8ABECFFFF            <1>       call   print_msg
4104 0000839A C3                    <1>       retn
4105                                <1>
4106                                <1> loc_set_time_stc_0:
4107                                <1>       ;xor   bh, bh ; video page 0
```

```
4108 0000839B E8AC9FFFFF          <1>       call   beeper ; BEEP !
4109 000083A0 E92EFEFFFF          <1>         jmp     loc_enter_hour_1
4110                              <1> loc_set_time_stc_1:
4111 000083A5 E8D1FDFFFF          <1>       call   check_for_backspace
4112 000083AA 740A                <1>       je     short loc_set_time_bs_1
4113                              <1>       ;xor   bh, bh ; video page 0
4114 000083AC E89B9FFFFF          <1>       call   beeper ; BEEP !
4115 000083B1 E950FEFFFF          <1>         jmp     loc_enter_hour_2
4116                              <1> loc_set_time_bs_1:
4117 000083B6 E8AEFDFFFF          <1>       call   write_backspace
4118 000083BB E913FEFFFF          <1>         jmp     loc_enter_hour_1
4119                              <1> loc_set_time_stc_2:
4120 000083C0 E8B6FDFFFF          <1>       call   check_for_backspace
4121 000083C5 740A                <1>       je     short loc_set_time_bs_2
4122                              <1>       ;xor   bh, bh ; video page 0
4123 000083C7 E8809FFFFF          <1>       call   beeper ; BEEP !
4124 000083CC E971FEFFFF          <1>         jmp     loc_enter_time_separator_1
4125                              <1> loc_set_time_bs_2:
4126 000083D1 E893FDFFFF          <1>       call   write_backspace
4127 000083D6 E92BFEFFFF          <1>         jmp     loc_enter_hour_2
4128                              <1> loc_set_time_stc_3:
4129 000083DB E89BFDFFFF          <1>       call   check_for_backspace
4130 000083E0 740A                <1>       je     short loc_set_time_bs_3
4131                              <1>       ;xor   bh, bh ; video page 0
4132 000083E2 E8659FFFFF          <1>       call   beeper ; BEEP !6
4133 000083E7 E974FEFFFF          <1>         jmp     loc_enter_minute_1
4134                              <1> loc_set_time_bs_3:
4135 000083EC E878FDFFFF          <1>       call   write_backspace
4136 000083F1 E94CFEFFFF          <1>         jmp     loc_enter_time_separator_1
4137                              <1> loc_set_time_stc_4:
4138 000083F6 E880FDFFFF          <1>       call   check_for_backspace
4139 000083FB 740A                <1>       je     short loc_set_time_bs_4
4140                              <1>       ;xor   bh, bh ; video page 0
4141 000083FD E84A9FFFFF          <1>       call   beeper ; BEEP !
4142 00008402 E984FEFFFF          <1>         jmp     loc_enter_minute_2
4143                              <1> loc_set_time_bs_4:
4144 00008407 E85DFDFFFF          <1>       call   write_backspace
4145 0000840C E94FFEFFFF          <1>         jmp     loc_enter_minute_1
4146                              <1> loc_set_time_stc_5:
4147 00008411 E865FDFFFF          <1>       call   check_for_backspace
4148 00008416 740A                <1>       je     short loc_set_time_bs_5
4149                              <1>       ;xor   bh, bh ; video page 0
4150 00008418 E82F9FFFFF          <1>       call   beeper ; BEEP !
4151 0000841D E994FEFFFF          <1>         jmp     loc_enter_time_separator_2
4152                              <1> loc_set_time_bs_5:
4153 00008422 E842FDFFFF          <1>       call   write_backspace
4154 00008427 E95FFEFFFF          <1>         jmp     loc_enter_minute_2
4155                              <1> loc_set_time_stc_6:
4156 0000842C E84AFDFFFF          <1>       call   check_for_backspace
4157 00008431 7413                <1>       je     short loc_set_time_bs_6
4158                              <1>       ;xor   bh, bh ; video page 0
4159 00008433 E8149FFFFF          <1>       call   beeper ; BEEP !
4160 00008438 66C705[C83A0100]30- <1>       mov    word [Second], 3030h
4160 00008440 30                  <1>
4161 00008441 E99FFEFFFF          <1>         jmp     loc_enter_second_1
4162                              <1> loc_set_time_bs_6:
4163 00008446 E81EFDFFFF          <1>       call   write_backspace
4164 0000844B E966FEFFFF          <1>         jmp     loc_enter_time_separator_2
4165                              <1> loc_set_time_stc_7:
4166 00008450 E826FDFFFF          <1>       call   check_for_backspace
4167 00008455 740A                <1>       je     short loc_set_time_bs_7
4168                              <1>       ;xor   bh, bh ; video page 0
4169 00008457 E8F09EFFFF          <1>       call   beeper ; BEEP !
4170 0000845C E9B3FEFFFF          <1>         jmp     loc_enter_second_2
4171                              <1> loc_set_time_bs_7:
4172 00008461 E803FDFFFF          <1>       call   write_backspace
4173 00008466 E97AFEFFFF          <1>         jmp     loc_enter_second_1
4174                              <1>
4175                              <1> print_volume_info:
4176                              <1>       ; 01/03/2016
4177                              <1>       ; 08/02/2016
4178                              <1>       ; 06/02/2016
4179                              <1>       ; 04/02/2016
4180                              <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
4181                              <1>       ; 25/10/2009
4182                              <1>       ;
4183                              <1>       ; "Volume Serial No: "
4184                              <1>       ;
4185                              <1>       ; INPUT  : AL = DOS Drive Number
4186                              <1>       ; OUTPUT : AH = FS Type
4187                              <1>       ;          AL = DOS Drive Name
4188                              <1>       ; CF = 0 -> OK
4189                              <1>       ; CF = 1 -> Drive not ready
4190                              <1>
4191 0000846B 88C4                <1>       mov    ah, al
4192 0000846D 28C0                <1>       sub    al, al
4193 0000846F 0FB7F0              <1>       movzx  esi, ax
4194 00008472 81C600010900        <1>       add    esi, Logical_DOSDisks
4195 00008478 8A06                <1>       mov    al, [esi]
4196 0000847A 3C41                <1>       cmp    al, 'A'
4197 0000847C 7304                <1>       jnb    short loc_pvi_set_vol_name
4198 0000847E 8A6604              <1>       mov    ah, [esi+LD_FSType]
4199 00008481 C3                  <1>       retn
4200                              <1>
4201                              <1> loc_pvi_set_vol_name:
4202 00008482 A2[FC3A0100]        <1>       mov    [Vol_Drv_Name], al
4203 00008487 56                  <1>       push   esi
4204 00008488 E858010000          <1>       call   move_volume_name_and_serial_no ;;;
4205 0000848D 7302                <1>       jnc    short loc_pvi_mvn_ok
4206 0000848F 5E                  <1>       pop    esi
4207 00008490 C3                  <1>       retn
4208                              <1>
4209                              <1> loc_pvi_mvn_ok:
4210 00008491 8B3424              <1>       mov    esi, [esp]
4211 00008494 807E04A1            <1>       cmp    byte [esi+LD_FSType], 0A1h
```

```
4212 00008498 7509              <1>       jne    short loc_pvi_fat_vol_size
4213 0000849A 8B4670            <1>       mov    eax, [esi+LD_FS_VolumeSize]
4214 0000849D 0FB75E11          <1>       movzx  ebx, word [esi+LD_FS_BytesPerSec]
4215 000084A1 EB07              <1>       jmp    short loc_vol_size_mul32
4216                            <1> loc_pvi_fat_vol_size:
4217 000084A3 8B4670            <1>       mov    eax, [esi+LD_TotalSectors]
4218 000084A6 0FB75E11          <1>       movzx  ebx, word [esi+LD_BPB+BPB_BytsPerSec]
4219                            <1> loc_vol_size_mul32:
4220 000084AA F7E3              <1>       mul    ebx
4221 000084AC 09D2              <1>       or     edx, edx
4222 000084AE 7507              <1>       jnz    short loc_vol_size_in_kbytes
4223                            <1> loc_vol_size_in_bytes:
4224 000084B0 B9[DA3A0100]      <1>       mov    ecx, VolSize_Bytes
4225 000084B5 EB0D              <1>       jmp    short loc_write_vol_size_str
4226                            <1> loc_vol_size_in_kbytes:
4227 000084B7 66BB0004          <1>       mov    bx, 1024
4228 000084BB F7F3              <1>       div    ebx
4229 000084BD B9[CD3A0100]      <1>       mov    ecx, VolSize_KiloBytes
4230 000084C2 31D2              <1>       xor    edx, edx ; 0
4231                            <1> loc_write_vol_size_str:
4232 000084C4 890D[A3880100]    <1>       mov    [VolSize_Unit1], ecx
4233                            <1>       ;
4234 000084CA BF[B9880100]      <1>       mov    edi, Vol_Tot_Sec_Str_End
4235                            <1>         ;mov byte [edi], 0
4236 000084CF B90A000000        <1>       mov    ecx, 10
4237                            <1> loc_write_vol_size_chr:
4238 000084D4 F7F1              <1>       div    ecx
4239 000084D6 80C230            <1>       add    dl, '0'
4240 000084D9 4F                <1>       dec    edi
4241 000084DA 8817              <1>       mov    [edi], dl
4242 000084DC 85C0              <1>       test   eax, eax
4243 000084DE 7404              <1>       jz     short loc_write_vol_size_str_ok
4244 000084E0 28D2              <1>       sub    dl, dl ; 0
4245 000084E2 EBF0              <1>       jmp    short loc_write_vol_size_chr
4246                            <1>
4247                            <1> loc_write_vol_size_str_ok:
4248 000084E4 893D[AB880100]    <1>       mov    [Vol_Tot_Sec_Str_Start], edi
4249                            <1>       ;
4250 000084EA BF[E53A0100]      <1>       mov    edi, Vol_FS_Name
4251 000084EF 8A4E03            <1>       mov    cl, [esi+LD_FATType]
4252 000084F2 20C9              <1>       and    cl, cl ; 0 ?
4253 000084F4 7515              <1>       jnz    short loc_write_vol_FAT_str_1
4254 000084F6 66C7075452        <1>       mov    word [edi], 'TR'
4255 000084FB C7470420465331    <1>       mov    dword [edi+4], ' FS1'
4256                            <1>       ;movzx ebx, word [esi+LD_FS_BytesPerSec]
4257 00008502 668B5E11          <1>       mov    bx, [esi+LD_FS_BytesPerSec]
4258 00008506 8B4674            <1>       mov    eax, [esi+LD_FS_FreeSectors]
4259 00008509 EB36              <1>       jmp    short loc_vol_freespace_mul32
4260                            <1>
4261                            <1> loc_write_vol_FAT_str_1:
4262 0000850B 66B83332          <1>       mov    ax, '32' ; FAT32
4263 0000850F 80F902            <1>       cmp    cl, 2 ; [esi+LD_FATType]
4264 00008512 7708              <1>       ja     short loc_write_vol_FAT_str_2
4265 00008514 66B83132          <1>       mov    ax, '12' ; FAT12
4266 00008518 7202              <1>       jb     short loc_write_vol_FAT_str_2
4267 0000851A B436              <1>       mov    ah, '6'  ; FAT16
4268                            <1> loc_write_vol_FAT_str_2:
4269 0000851C C70746415420      <1>       mov    dword [edi], 'FAT '
4270 00008522 66894704          <1>       mov    word [edi+4], ax
4271                            <1>       ;
4272                            <1>       ;movzx ebx, word [esi+LD_BPB+BPB_BytsPerSec]
4273 00008526 668B5E11          <1>       mov    bx, [esi+LD_BPB+BPB_BytsPerSec]
4274 0000852A 8B4674            <1>       mov    eax, [esi+LD_FreeSectors]
4275                            <1>
4276                            <1> loc_vol_freespace_recalc0:
4277                            <1>       ; 01/03/2016
4278 0000852D 83F8FF            <1>       cmp    eax, 0FFFFFFFFh
4279 00008530 720F              <1>       jb     short loc_vol_freespace_mul32
4280                            <1>       ;inc   eax ; 0
4281 00008532 20C9              <1>       and    cl, cl ; byte [esi+LD_FATType]
4282 00008534 740B              <1>       jz     short loc_vol_freespace_mul32
4283 00008536 53                <1>       push   ebx
4284 00008537 66BB00FF          <1>       mov    bx, 0FF00h ; recalculate free sectors
4285 0000853B E876490000        <1>       call   calculate_fat_freespace
4286 00008540 5B                <1>       pop    ebx
4287                            <1>
4288                            <1> loc_vol_freespace_mul32:
4289 00008541 F7E3              <1>       mul    ebx
4290 00008543 09D2              <1>       or     edx, edx
4291 00008545 7507              <1>       jnz    short loc_vol_fspace_in_kbytes
4292                            <1> loc_vol_fspace_in_bytes:
4293 00008547 B9[DA3A0100]      <1>       mov    ecx, VolSize_Bytes
4294 0000854C EB0D              <1>       jmp    short loc_write_vol_fspace_str
4295                            <1> loc_vol_fspace_in_kbytes:
4296 0000854E 66BB0004          <1>       mov    bx, 1024
4297 00008552 F7F3              <1>       div    ebx
4298 00008554 B9[CD3A0100]      <1>       mov    ecx, VolSize_KiloBytes
4299 00008559 31D2              <1>       xor    edx, edx ; 0
4300                            <1> loc_write_vol_fspace_str:
4301 0000855B 890D[A7880100]    <1>       mov    [VolSize_Unit2], ecx
4302                            <1>       ;
4303 00008561 BF[C9880100]      <1>       mov    edi, Vol_Free_Sectors_Str_End
4304                            <1>         ;mov byte [edi], 0
4305 00008566 B90A000000        <1>       mov    ecx, 10
4306                            <1> loc_write_vol_fspace_chr:
4307 0000856B F7F1              <1>       div    ecx
4308 0000856D 80C230            <1>       add    dl, '0'
4309 00008570 4F                <1>       dec    edi
4310 00008571 8817              <1>       mov    [edi], dl
4311 00008573 85C0              <1>       test   eax, eax
4312 00008575 7404              <1>       jz     short loc_write_vol_fspace_str_ok
4313 00008577 28D2              <1>       sub    dl, dl ; 0
4314 00008579 EBF0              <1>       jmp    short loc_write_vol_fspace_chr
4315                            <1>
4316                            <1> loc_write_vol_fspace_str_ok:
```

```
4317 0000857B 893D[BB880100]      <1>       mov   [Vol_Free_Sectors_Str_Start], edi
4318                              <1>       ;
4319 00008581 BE[E33A0100]        <1>       mov   esi, Volume_in_drive
4320 00008586 E8BAEAFFFF          <1>       call  print_msg
4321 0000858B BE[233B0100]        <1>       mov   esi, Vol_Name
4322 00008590 E8B0EAFFFF          <1>       call  print_msg
4323 00008595 BE[D7430100]        <1>       mov   esi, nextline
4324 0000859A E8A6EAFFFF          <1>       call  print_msg
4325                              <1>       ;
4326 0000859F BE[843B0100]        <1>       mov   esi, Vol_Total_Sector_Header
4327 000085A4 E89CEAFFFF          <1>       call  print_msg
4328 000085A9 8B35[AB880100]      <1>       mov   esi, [Vol_Tot_Sec_Str_Start]
4329 000085AF E891EAFFFF          <1>       call  print_msg
4330 000085B4 8B35[A3880100]      <1>       mov   esi, [VolSize_Unit1]
4331 000085BA E886EAFFFF          <1>       call  print_msg
4332                              <1>       ;
4333 000085BF BE[953B0100]        <1>       mov   esi, Vol_Free_Sectors_Header
4334 000085C4 E87CEAFFFF          <1>       call  print_msg
4335 000085C9 8B35[BB880100]      <1>       mov   esi, [Vol_Free_Sectors_Str_Start]
4336 000085CF E871EAFFFF          <1>       call  print_msg
4337 000085D4 8B35[A7880100]      <1>       mov   esi, [VolSize_Unit2]
4338 000085DA E866EAFFFF          <1>       call  print_msg
4339                              <1>       ;
4340 000085DF 5E                  <1>       pop   esi
4341                              <1>
4342                              <1>       ;mov  ah, [esi+LD_FSType]
4343                              <1>       ;mov  al, [esi+LD_FATType]
4344 000085E0 668B4603            <1>       mov   ax, [esi+LD_FATType]
4345                              <1>
4346 000085E4 C3                  <1>       retn
4347                              <1>
4348                              <1> move_volume_name_and_serial_no:
4349                              <1>       ; 08/02/2016  (TRDOS 386 = TRDOS v2.0)
4350                              <1>       ; this routine will be called by
4351                              <1>       ; "print_volume_info" and "print_directory"
4352                              <1>       ; INPUT ->
4353                              <1>       ;     ESI = Logical DOS drv descripton table address
4354                              <1>       ; OUTPUT ->
4355                              <1>       ;     *Volume name will be moved to text area
4356                              <1>       ;     *Volume serial number will be converted to
4357                              <1>       ;      text and will be moved to text area
4358                              <1>       ;   cf = 1 -> invalid/unknown dos drive
4359                              <1>       ;   cf = 0 -> ecx = 0
4360                              <1>       ;
4361                              <1>       ; (eax, edx, ecx, esi, edi will be changed)
4362                              <1>
4363 000085E5 BF[233B0100]        <1>       mov   edi, Vol_Name
4364                              <1>
4365                              <1>       ;mov  ah, [esi+LD_FSType]
4366                              <1>       ;mov  al, [esi+LD_FATType]
4367 000085EA 668B4603            <1>       mov   ax, [esi+LD_FATType]
4368 000085EE 80FCA1              <1>       cmp   ah, 0A1h
4369 000085F1 7418                <1>       je    short mvn_2
4370 000085F3 08E4                <1>       or    ah, ah
4371 000085F5 7404                <1>       jz    short mvn_0
4372 000085F7 08C0                <1>       or    al, al
4373 000085F9 7504                <1>       jnz   short mvn_1
4374                              <1> mvn_0:
4375 000085FB 8A06                <1>       mov   al, [esi]
4376 000085FD F9                  <1>       stc
4377 000085FE C3                  <1>       retn
4378                              <1> mvn_1:
4379 000085FF 3C02                <1>       cmp   al, 2
4380 00008601 7717                <1>       ja    short mvn_3
4381                              <1>       ;or   al, al
4382                              <1>       ;jz   short mvn_2
4383 00008603 8B462D              <1>       mov   eax, [esi+LD_BPB+VolumeID]
4384 00008606 83C631              <1>       add   esi, LD_BPB+VolumeLabel
4385 00008609 EB15                <1>       jmp   short mvn_4
4386                              <1> mvn_2:
4387 0000860B 8B4628              <1>       mov   eax, [esi+LD_FS_VolumeSerial]
4388 0000860E 83C62C              <1>       add   esi, LD_FS_VolumeName
4389 00008611 B910000000          <1>       mov   ecx, 16
4390 00008616 F3A5                <1>       rep   movsd
4391 00008618 EB10                <1>       jmp   short mvn_5
4392                              <1> mvn_3:
4393 0000861A 8B4649              <1>       mov   eax, [esi+LD_BPB+FAT32_VolID]
4394 0000861D 83C64D              <1>       add   esi, LD_BPB+FAT32_VolLab
4395                              <1> mvn_4:
4396 00008620 B90B000000          <1>       mov   ecx, 11
4397 00008625 F3A4                <1>       rep   movsb
4398 00008627 C60700              <1>       mov   byte [edi], 0
4399                              <1> mvn_5:
4400                              <1>       ;mov  [Current_VolSerial], eax
4401 0000862A E820BCFFFF          <1>       call  dwordtohex
4402 0000862F 8915[783B0100]      <1>       mov   [Vol_Serial1], edx
4403 00008635 A3[7D3B0100]        <1>       mov   [Vol_Serial2], eax
4404                              <1>       ; ecx = 0
4405 0000863A C3                  <1>       retn
4406                              <1>
4407                              <1> get_volume_serial_number:
4408                              <1>       ; 19/01/2016 (TRDOS 386 = TRDOS v2.0)
4409                              <1>       ; 08/08/2010
4410                              <1>       ;
4411                              <1>       ; INPUT -> DL = Logical DOS Drive number
4412                              <1>       ; OUTPUT -> EAX = Volume serial number
4413                              <1>       ;          BL= FAT Type
4414                              <1>       ;          BH = Logical DOS drv Number (DL input)
4415                              <1>       ; cf = 1 -> Drive not ready
4416                              <1>
4417 0000863B 31DB                <1>       xor   ebx, ebx
4418 0000863D 88D7                <1>       mov   bh, dl
4419 0000863F 3815[54390100]      <1>       cmp   [Last_DOS_DiskNo], dl
4420 00008645 7304                <1>       jnb   short loc_gvsn_start
4421                              <1> loc_gvsn_stc_retn:
```

```
4422 00008647 31C0                <1>       xor    eax, eax
4423 00008649 F9                  <1>       stc
4424 0000864A C3                  <1>         retn
4425                              <1> loc_gvsn_start:
4426 0000864B 56                  <1>       push   esi
4427 0000864C BE00010900          <1>       mov    esi, Logical_DOSDisks
4428 00008651 01DE                <1>       add    esi, ebx
4429 00008653 8A5E03              <1>       mov    bl, [esi+LD_FATType]
4430 00008656 20DB                <1>       and    bl, bl
4431 00008658 740F                <1>       jz     short loc_gvsn_fs
4432 0000865A 80FB02              <1>       cmp    bl, 2
4433 0000865D 7705                <1>       ja     short loc_gvsn_fat32
4434                              <1> loc_gvsn_fat:
4435 0000865F 83C62D              <1>       add    esi, LD_BPB + VolumeID
4436 00008662 EB0E                <1>       jmp    short loc_gvsn_return
4437                              <1> loc_gvsn_fat32:
4438 00008664 83C649              <1>       add    esi, LD_BPB + FAT32_VolID
4439 00008667 EB09                <1>       jmp    short loc_gvsn_return
4440                              <1> loc_gvsn_fs:
4441 00008669 807E04A1            <1>       cmp    byte [esi+LD_FSType], 0A1h
4442 0000866D 75D8                <1>       jne    short loc_gvsn_stc_retn
4443 0000866F 83C628              <1>       add    esi, LD_FS_VolumeSerial
4444                              <1> loc_gvsn_return:
4445 00008672 8B06                <1>       mov    eax, [esi]
4446 00008674 5E                  <1>       pop    esi
4447 00008675 C3                  <1>       retn
4448                              <1>
4449                              <1> ; CMD_INTR.ASM [ TRDOS Command Interpreter Procedure ]
4450                              <1> ; 09/11/2011
4451                              <1> ; 29/01/2005
4452                              <1>
4453                              <1> command_interpreter:
4454                              <1>       ; 16/10/2016
4455                              <1>       ; 12/10/2016
4456                              <1>       ; 13/05/2016
4457                              <1>       ; 07/05/2016
4458                              <1>       ; 04/03/2016
4459                              <1>       ; 04/02/2016
4460                              <1>       ; 03/02/2016
4461                              <1>       ; 30/01/2016
4462                              <1>       ; 29/01/2016 (TRDOS 386 = TRDOS 2.0)
4463                              <1>       ; 15/09/2011
4464                              <1>       ; 29/01/2005
4465                              <1>
4466                              <1>       ; Input: ecx = command word length (CL)
4467                              <1>       ;       CommandBuffer = Command string offset
4468                              <1>
4469 00008676 C605[5C890100]00    <1>       mov    byte [Program_Exit],0
4470 0000867D 80F904              <1>       cmp    cl, 4
4471 00008680 0F87B5020000        <1>         ja      c_6
4472 00008686 0F8237010000        <1>         jb      c_2
4473                              <1> c_4:
4474                              <1>
4475                              <1> cmp_cmd_exit:
4476 0000868C BF[C2390100]        <1>       mov    edi, Cmd_Exit
4477 00008691 E8C2030000          <1>       call   cmp_cmd
4478 00008696 7208                <1>       jc     short cmp_cmd_date
4479                              <1>
4480 00008698 C605[5C890100]01    <1>         mov     byte [Program_Exit], 1
4481 0000869F C3                  <1>         retn
4482                              <1>
4483                              <1> cmp_cmd_date:
4484 000086A0 B104                <1>       mov    cl, 4
4485 000086A2 BF[DE390100]        <1>       mov    edi, Cmd_Date
4486 000086A7 E8AC030000          <1>       call   cmp_cmd
4487 000086AC 720B                <1>         jc   short cmp_cmd_time
4488                              <1>
4489 000086AE E8D0F7FFFF          <1>       call   show_date
4490 000086B3 E80FF8FFFF          <1>       call   set_date
4491 000086B8 C3                  <1>       retn
4492                              <1>
4493                              <1> cmp_cmd_time:
4494 000086B9 B104                <1>       mov    cl, 4
4495 000086BB BF[E3390100]        <1>       mov    edi, Cmd_Time
4496 000086C0 E893030000          <1>       call   cmp_cmd
4497 000086C5 720B                <1>       jc     short cmp_cmd_show
4498                              <1>
4499 000086C7 E8C6FAFFFF          <1>       call   show_time
4500 000086CC E8F8FAFFFF          <1>       call   set_time
4501 000086D1 C3                  <1>       retn
4502                              <1>
4503                              <1> cmp_cmd_show:
4504 000086D2 B104                <1>       mov    cl, 4
4505 000086D4 BF[F4390100]        <1>       mov    edi, Cmd_Show
4506 000086D9 E87A030000          <1>       call   cmp_cmd
4507 000086DE 0F83050A0000        <1>         jnc     show_file
4508                              <1>
4509                              <1> cmp_cmd_echo:
4510 000086E4 B104                <1>       mov    cl, 4
4511 000086E6 BF[303A0100]        <1>       mov    edi, Cmd_Echo
4512 000086EB E868030000          <1>       call   cmp_cmd
4513 000086F0 7224                <1>       jc     short cmp_cmd_copy
4514                              <1>
4515                              <1>       ; 22/11/2017
4516                              <1>       ; AL = 0
4517 000086F2 803E20              <1>       cmp    byte [esi], 20h
4518 000086F5 7215                <1>       jb     short cmd_echo_nextline
4519                              <1>       ; 14/04/2016
4520 000086F7 56                  <1>       push   esi
4521                              <1> cmd_echo_asciiz:
4522                              <1>       ;inc   esi
4523                              <1>       ;mov   al, [esi]
4524                              <1>       ; 22/11/2017
4525 000086F8 AC                  <1>       lodsb
4526 000086F9 3C20                <1>       cmp    al, 20h
```

```
4527 000086FB 73FB              <1>       jnb    short cmd_echo_asciiz
4528 000086FD 4E                <1>       dec    esi
4529 000086FE C60600            <1>       mov    byte [esi], 0
4530 00008701 5E                <1>       pop    esi
4531 00008702 89F7              <1>       mov    edi, esi
4532 00008704 E83CE9FFFF        <1>       call   print_msg
4533 00008709 C60700            <1>       mov    byte [edi], 0
4534                            <1> cmd_echo_nextline:
4535 0000870C BE[45440100]      <1>       mov    esi, NextLine
4536                            <1>       ;call  print_msg
4537                            <1>       ;retn
4538 00008711 E92FE9FFFF        <1>       jmp    print_msg
4539                            <1>
4540                            <1> cmp_cmd_copy:
4541 00008716 B104              <1>       mov    cl, 4
4542 00008718 BF[173A0100]      <1>       mov    edi, Cmd_Copy
4543 0000871D E836030000        <1>       call   cmp_cmd
4544 00008722 0F83CC170000      <1>       jnc    copy_file
4545                            <1>
4546                            <1> cmp_cmd_move:
4547 00008728 B104              <1>       mov    cl, 4
4548 0000872A BF[1C3A0100]      <1>       mov    edi, Cmd_Move
4549 0000872F E824030000        <1>       call   cmp_cmd
4550 00008734 0F836E160000      <1>       jnc    move_file
4551                            <1>
4552                            <1> cmp_cmd_path:
4553 0000873A B104              <1>       mov    cl, 4
4554 0000873C BF[213A0100]      <1>       mov    edi, Cmd_Path
4555 00008741 E812030000        <1>       call   cmp_cmd
4556 00008746 0F83F0190000      <1>       jnc    set_get_path
4557                            <1>
4558                            <1> cmp_cmd_beep:
4559 0000874C B104              <1>       mov    cl, 4
4560 0000874E BF[4E3A0100]      <1>       mov    edi, Cmd_Beep
4561 00008753 E800030000        <1>       call   cmp_cmd
4562 00008758 720B              <1>       jc     short cmp_cmd_find
4563                            <1>       ; 13/05/2016
4564 0000875A 8A3D[D6800100]    <1>       mov    bh, [ptty] ; [ACTIVE_PAGE]
4565 00008760 E9E79BFFFF        <1>       jmp    beeper
4566                            <1>
4567                            <1> cmp_cmd_find:
4568 00008765 B104              <1>       mov    cl, 4
4569 00008767 BF[2B3A0100]      <1>       mov    edi, Cmd_Find
4570 0000876C E8E7020000        <1>       call   cmp_cmd
4571 00008771 0F82C4020000      <1>         jc     cmp_cmd_external
4572                            <1>
4573                            <1>       ;call  find_and_list_files
4574 00008777 E9AF220000        <1>       jmp    find_and_list_files
4575                            <1>       ;retn
4576                            <1>
4577                            <1> c_1:
4578 0000877C AD                <1>       lodsd
4579                            <1> cmp_cmd_help:
4580 0000877D 3C3F              <1>       cmp    al, '?'
4581 0000877F 751D              <1>         jne    short cmp_cmd_remark
4582                            <1>
4583 00008781 BE[B4390100]      <1>       mov    esi, Command_List
4584                            <1> cmd_help_next_w:
4585 00008786 E8BAE8FFFF        <1>       call   print_msg
4586                            <1>
4587 0000878B 803E20            <1>       cmp    byte [esi], 20h ; 0
4588 0000878E 7232              <1>       jb     short cmd_help_retn
4589                            <1>
4590 00008790 56                <1>       push   esi
4591 00008791 BE[D7430100]      <1>       mov    esi, nextline
4592 00008796 E8AAE8FFFF        <1>       call   print_msg
4593 0000879B 5E                <1>       pop    esi
4594 0000879C EBE8              <1>       jmp    short cmd_help_next_w
4595                            <1>
4596                            <1> cmp_cmd_remark:
4597 0000879E 3C2A              <1>       cmp    al, '*'
4598 000087A0 0F8595020000      <1>         jne    cmp_cmd_external
4599 000087A6 46                <1>       inc    esi
4600 000087A7 BF[D0810100]      <1>       mov    edi, Remark
4601 000087AC 8A06              <1>       mov    al, [esi]
4602 000087AE 3C20              <1>       cmp    al, 20h
4603 000087B0 7707              <1>       ja     short cmd_remark_write
4604 000087B2 89FE              <1>       mov    esi, edi ; Remark
4605 000087B4 E98CE8FFFF        <1>       jmp    print_msg
4606                            <1>
4607                            <1> cmd_remark_write:
4608 000087B9 AA                <1>       stosb
4609 000087BA AC                <1>       lodsb
4610 000087BB 3C20              <1>       cmp    al, 20h
4611 000087BD 73FA              <1>       jnb    short cmd_remark_write
4612 000087BF C60700            <1>       mov    byte [edi], 0
4613                            <1>
4614                            <1> cmd_help_retn:
4615                            <1> cmd_remark_retn:
4616                            <1> cd_retn:
4617 000087C2 C3                <1>       retn
4618                            <1>
4619                            <1> c_2:
4620 000087C3 80F902            <1>       cmp    cl, 2
4621 000087C6 0F87AF000000      <1>         ja     c_3
4622 000087CC BE[1E820100]      <1>       mov    esi, CommandBuffer
4623 000087D1 72A9              <1>       jb     short c_1
4624                            <1>
4625                            <1> cmp_cmd_cd:
4626 000087D3 66AD              <1>       lodsw
4627 000087D5 663D4344          <1>       cmp    ax, 'CD'
4628 000087D9 7551              <1>       jne    short cmp_cmd_drive
4629 000087DB 46                <1>         inc esi
4630                            <1> cd_0:
4631 000087DC 668B06            <1>       mov    ax, [esi]
```

```
4632 000087DF 3C20                <1>       cmp    al, 20h
4633 000087E1 76DF                <1>       jna    short cd_retn
4634                              <1>       ; 10/02/2016
4635 000087E3 80FC3A              <1>       cmp    ah, ':'
4636 000087E6 7504                <1>       jne    short cd_1
4637 000087E8 46                  <1>       inc    esi
4638 000087E9 46                  <1>       inc    esi
4639 000087EA EB49                <1>       jmp    short cd_2
4640                              <1>
4641                              <1> cd_1: ; change current directory
4642                              <1>       ; 29/11/2009
4643                              <1>       ; AH = CDh    ; to separate 'CD' command from others
4644                              <1>                     ; for restoring current directory
4645                              <1>                     ; 0CDh sign is for saving cdir into
4646                              <1>                     ; DOS drv description table cdir area
4647                              <1>
4648 000087EC B4CD                <1>       mov    ah, 0CDh ; mov byte [CD_COMMAND], 0CDh
4649                              <1>
4650 000087EE E81D230000          <1>       call   change_current_directory
4651 000087F3 0F8337220000        <1>         jnc     change_prompt_dir_string
4652                              <1>
4653                              <1> cd_error_messages:
4654 000087F9 3C03                <1>       cmp    al, 3
4655 000087FB 740C                <1>       je     short cd_path_not_found
4656                              <1>       ; 16/10/2016 (15h -> 15)
4657 000087FD 3C0F                <1>       cmp    al, 15 ; drive not ready error
4658 000087FF 7459                <1>       je     short cd_drive_not_ready
4659 00008801 3C11                <1>       cmp    al, 17 ; read error
4660 00008803 7455                <1>       je     short cd_drive_not_ready
4661 00008805 3C13                <1>       cmp    al, 19 ; ; Bad directory/path name
4662 00008807 7466                <1>       je     short cd_command_failed
4663                              <1>
4664                              <1> cd_path_not_found:
4665 00008809 50                  <1>       push   eax ; 29/12/2017
4666                              <1>       ;push  ax
4667 0000880A BE[573C0100]        <1>       mov    esi, Msg_Dir_Not_Found
4668 0000880F E831E8FFFF          <1>       call   print_msg
4669                              <1>       ;pop   ax
4670 00008814 58                  <1>       pop    eax ; 29/12/2017
4671 00008815 3A25[6C810100]      <1>       cmp    ah, [Current_Dir_Level]
4672 0000881B 0F830F220000        <1>         jnb     change_prompt_dir_string
4673 00008821 8825[6C810100]      <1>       mov    [Current_Dir_Level], ah
4674 00008827 E904220000          <1>         jmp     change_prompt_dir_string
4675                              <1>
4676                              <1> cmp_cmd_drive: ; change current drive
4677                              <1>       ; C:, D:, E: etc.
4678 0000882C 80FC3A              <1>       cmp    ah, ':'
4679 0000882F 0F8506020000        <1>         jne     cmp_cmd_external
4680                              <1>
4681                              <1> cd_2: ; 'CD C:', 'CD D:' ...
4682 00008835 803E20              <1>       cmp    byte [esi], 20h
4683 00008838 0F8707020000        <1>         ja      loc_cmd_failed
4684                              <1>
4685 0000883E 24DF                <1>       and    al, 0DFh
4686 00008840 2C41                <1>       sub    al, 'A'
4687 00008842 0F82FD010000        <1>         jc      loc_cmd_failed
4688                              <1>
4689 00008848 3A05[54390100]      <1>         cmp     al, [Last_DOS_DiskNo]
4690 0000884E 770A                <1>         ja   short cd_drive_not_ready
4691                              <1>
4692 00008850 88C2                <1>       mov    dl, al
4693 00008852 E85BF3FFFF          <1>       call   change_current_drive
4694 00008857 7201                <1>       jc     short cd_drive_not_ready
4695 00008859 C3                  <1>       retn
4696                              <1>
4697                              <1> cd_drive_not_ready:
4698 0000885A BE[143C0100]        <1>       mov    esi, Msg_Not_Ready_Read_Err
4699 0000885F E8E1E7FFFF          <1>       call   print_msg
4700                              <1>
4701                              <1> cd_fail_drive_restart:
4702 00008864 8A15[6E810100]      <1>       mov    dl, [Current_Drv]
4703                              <1>       ;call  change_current_drive
4704 0000886A E943F3FFFF          <1>         jmp     change_current_drive
4705                              <1>       ;retn
4706                              <1>
4707                              <1> cd_command_failed:
4708 0000886F BE[F53B0100]        <1>       mov    esi, Msg_Bad_Command
4709 00008874 E8CCE7FFFF          <1>       call   print_msg
4710 00008879 EBE9                <1>       jmp    short cd_fail_drive_restart
4711                              <1>
4712                              <1> c_3:
4713                              <1> cmp_cmd_dir:
4714 0000887B BF[B4390100]        <1>       mov    edi, Cmd_Dir
4715 00008880 E8D3010000          <1>       call   cmp_cmd
4716 00008885 0F8380020000        <1>       jnc    print_directory_list
4717                              <1>
4718                              <1> cmp_cmd_cls:
4719 0000888B B103                <1>       mov    cl, 3
4720 0000888D BF[F0390100]        <1>       mov    edi, Cmd_Cls
4721 00008892 E8C1010000          <1>       call   cmp_cmd
4722 00008897 0F83BEE7FFFF        <1>         jnc  clear_screen
4723                              <1>
4724                              <1> cmp_cmd_ver:
4725 0000889D B103                <1>       mov    cl, 3
4726 0000889F BF[BE390100]        <1>       mov    edi, Cmd_Ver
4727 000088A4 E8AF010000          <1>       call   cmp_cmd
4728 000088A9 720A                <1>       jc     short cmp_cmd_mem
4729                              <1>
4730 000088AB BE[5C390100]        <1>       mov    esi, mainprog_Version
4731                              <1>       ;call  print_msg
4732 000088B0 E990E7FFFF          <1>       jmp    print_msg
4733                              <1>       ;retn
4734                              <1>
4735                              <1> cmp_cmd_mem:
4736 000088B5 B103                <1>       mov    cl, 3
```

```
4737 000088B7 BF[263A0100]        <1>       mov    edi, Cmd_Mem
4738 000088BC E897010000          <1>       call   cmp_cmd
4739 000088C1 0F83BAB8FFFF        <1>       jnc    memory_info
4740                              <1>
4741                              <1> cmp_cmd_del:
4742 000088C7 B103                <1>       mov    cl, 3
4743 000088C9 BF[F9390100]        <1>       mov    edi, Cmd_Del
4744 000088CE E885010000          <1>       call   cmp_cmd
4745 000088D3 0F83280F0000        <1>         jnc     delete_file
4746                              <1>
4747                              <1> cmp_cmd_set:
4748 000088D9 B103                <1>       mov    cl, 3
4749 000088DB BF[EC390100]        <1>       mov    edi, Cmd_Set
4750 000088E0 E873010000          <1>       call   cmp_cmd
4751 000088E5 0F83C9170000        <1>         jnc     set_get_env
4752                              <1>
4753                              <1> cmp_cmd_run:
4754 000088EB B103                <1>       mov    cl, 3
4755 000088ED BF[E8390100]        <1>       mov    edi, Cmd_Run
4756 000088F2 E861010000          <1>       call   cmp_cmd
4757                              <1>       ; 07/05/2016
4758 000088F7 0F823E010000        <1>         jc      cmp_cmd_external
4759 000088FD E90F1E0000          <1>       jmp    load_and_execute_file
4760                              <1> c_5:
4761                              <1> cmp_cmd_mkdir:
4762 00008902 BF[113A0100]        <1>       mov    edi, Cmd_Mkdir
4763 00008907 E84C010000          <1>       call   cmp_cmd
4764 0000890C 0F83990A0000        <1>         jnc     make_directory
4765                              <1>
4766                              <1> cmp_cmd_rmdir:
4767 00008912 B105                <1>       mov    cl, 5
4768 00008914 BF[0B3A0100]        <1>       mov    edi, Cmd_Rmdir
4769 00008919 E83A010000          <1>       call   cmp_cmd
4770 0000891E 0F83AA0B0000        <1>         jnc     delete_directory
4771                              <1>
4772                              <1> cmp_cmd_chdir:
4773 00008924 B105                <1>       mov    cl, 5
4774 00008926 BF[483A0100]        <1>       mov    edi, Cmd_Chdir
4775 0000892B E828010000          <1>       call   cmp_cmd
4776 00008930 0F8205010000        <1>         jc      cmp_cmd_external
4777                              <1>
4778 00008936 E9A1FEFFFF          <1>       jmp    cd_0
4779                              <1>
4780                              <1> c_6:
4781 0000893B 80F906              <1>       cmp    cl, 6
4782 0000893E 0F87E0000000        <1>         ja      c_8
4783 00008944 72BC                <1>       jb     short c_5
4784                              <1> cmp_cmd_prompt:
4785 00008946 BF[C7390100]        <1>       mov    edi, Cmd_Prompt
4786 0000894B E808010000          <1>       call   cmp_cmd
4787 00008950 722F                <1>         jc   short cmp_cmd_volume
4788                              <1> get_prompt_name_fchar:
4789 00008952 AC                  <1>       lodsb
4790 00008953 3C20                <1>       cmp    al, 20h
4791 00008955 74FB                <1>       je     short get_prompt_name_fchar
4792 00008957 7713                <1>       ja     short loc_change_prompt_label
4793                              <1> default_command_prompt: ; 31/12/2017 ('sysprompt')
4794 00008959 BE[A8390100]        <1>       mov    esi, TRDOSPromptLabel
4795 0000895E C7065452444F        <1>       mov    dword [esi], "TRDO"
4796 00008964 66C746045300        <1>              mov    word [esi+4], "S"
4797                              <1> loc_cmd_prompt_return:
4798 0000896A C3                  <1>       retn
4799                              <1>
4800                              <1> set_command_prompt: ; 31/12/2017 ('sysprompt')
4801 0000896B AC                  <1>       lodsb
4802                              <1> loc_change_prompt_label:
4803 0000896C 66B90B00            <1>       mov    cx, 11
4804 00008970 BF[A8390100]        <1>       mov    edi, TRDOSPromptLabel
4805                              <1> put_char_new_prompt_label:
4806 00008975 AA                  <1>       stosb
4807 00008976 AC                  <1>       lodsb
4808 00008977 3C20                <1>       cmp    al, 20h
4809 00008979 7202                <1>       jb     short pass_put_new_prompt_label
4810 0000897B E2F8                <1>       loop   put_char_new_prompt_label
4811                              <1> pass_put_new_prompt_label:
4812 0000897D C60700              <1>       mov    byte [edi], 0
4813 00008980 C3                  <1>       retn
4814                              <1>
4815                              <1> cmp_cmd_volume:
4816 00008981 B106                <1>       mov    cl, 6
4817 00008983 BF[CE390100]        <1>       mov    edi, Cmd_Volume
4818 00008988 E8CB000000          <1>       call   cmp_cmd
4819 0000898D 7255                <1>         jc   short cmp_cmd_attrib
4820                              <1>
4821                              <1> cmd_vol1:
4822 0000898F AC                  <1>       lodsb
4823 00008990 3C20                <1>       cmp    al, 20h
4824 00008992 7707                <1>       ja     short cmd_vol2
4825 00008994 A0[6E810100]        <1>       mov    al, [Current_Drv]
4826 00008999 EB3D                <1>       jmp    short cmd_vol4
4827                              <1> cmd_vol2:
4828 0000899B 3C41                <1>       cmp    al, 'A'
4829 0000899D 0F82A2000000        <1>         jb      loc_cmd_failed
4830 000089A3 3C7A                <1>       cmp    al, 'z'
4831 000089A5 0F879A000000        <1>         ja      loc_cmd_failed
4832 000089AB 3C5A                <1>       cmp    al, 'Z'
4833 000089AD 760A                <1>       jna    short cmd_vol3
4834 000089AF 3C61                <1>       cmp    al, 'a'
4835 000089B1 0F828E000000        <1>         jb      loc_cmd_failed
4836 000089B7 24DF                <1>       and    al, 0DFh
4837                              <1> cmd_vol3:
4838 000089B9 8A26                <1>       mov    ah, [esi]
4839 000089BB 80FC3A              <1>       cmp    ah, ':'
4840 000089BE 0F8581000000        <1>         jne     loc_cmd_failed
4841 000089C4 2C41                <1>       sub    al, 'A'
```

```
4842 000089C6 3A05[54390100]      <1>         cmp     al, [Last_DOS_DiskNo]
4843 000089CC 760A                <1>       jna    short cmd_vol4
4844                              <1>
4845 000089CE BE[143C0100]        <1>       mov    esi, Msg_Not_Ready_Read_Err
4846 000089D3 E96DE6FFFF          <1>       jmp    print_msg
4847                              <1>
4848                              <1> cmd_vol4:
4849 000089D8 E88EFAFFFF          <1>       call   print_volume_info
4850 000089DD 0F8277FEFFFF        <1>         jc      cd_drive_not_ready
4851 000089E3 C3                  <1>       retn
4852                              <1>
4853                              <1> cmp_cmd_attrib:
4854 000089E4 B106                <1>       mov    cl, 6
4855 000089E6 BF[FD390100]        <1>       mov    edi, Cmd_Attrib
4856 000089EB E868000000          <1>       call   cmp_cmd
4857 000089F0 0F831D0F0000        <1>         jnc     set_file_attributes
4858                              <1>
4859                              <1> cmp_cmd_rename:
4860 000089F6 B106                <1>       mov    cl, 6
4861 000089F8 BF[043A0100]        <1>       mov    edi, Cmd_Rename
4862 000089FD E856000000          <1>       call   cmp_cmd
4863 00008A02 0F8353110000        <1>         jnc     rename_file
4864                              <1>
4865                              <1> cmp_cmd_device:
4866 00008A08 B106                <1>       mov    cl, 6
4867 00008A0A BF[393A0100]        <1>       mov    edi, Cmd_Device
4868 00008A0F E844000000          <1>       call   cmp_cmd
4869 00008A14 7225                <1>         jc   short cmp_cmd_external
4870                              <1>
4871 00008A16 C3                  <1>       retn
4872                              <1>
4873                              <1> c_7:
4874                              <1> cmp_cmd_devlist:
4875 00008A17 BF[403A0100]        <1>       mov    edi, Cmd_DevList
4876 00008A1C E837000000          <1>       call   cmp_cmd
4877 00008A21 7218                <1>         jc   short cmp_cmd_external
4878                              <1>
4879                              <1> loc_cmd_return:
4880 00008A23 C3                  <1>       retn
4881                              <1>
4882                              <1> c_8:
4883 00008A24 80F908              <1>         cmp  cl, 8
4884 00008A27 7712                <1>       ja     short cmp_cmd_external
4885 00008A29 72EC                <1>       jb     short c_7
4886                              <1>
4887                              <1> cmp_cmd_longname:
4888 00008A2B BF[D5390100]        <1>       mov    edi, Cmd_LongName
4889 00008A30 E823000000          <1>       call   cmp_cmd
4890 00008A35 0F8350060000        <1>         jnc     get_and_print_longname
4891                              <1>
4892                              <1> cmp_cmd_external:
4893                              <1>       ; 07/05/2016
4894                              <1>       ; 22/04/2016
4895 00008A3B BE[1E820100]        <1>       mov    esi, CommandBuffer
4896 00008A40 E9CC1C0000          <1>       jmp    loc_run_check_filename
4897                              <1>
4898                              <1> loc_cmd_failed:
4899 00008A45 803D[1E820100]20    <1>       cmp    byte [CommandBuffer], 20h
4900 00008A4C 76D5                <1>       jna    short loc_cmd_return
4901 00008A4E BE[F53B0100]        <1>       mov    esi, Msg_Bad_Command
4902                              <1> ;     call   print_msg
4903                              <1> ;loc_cmd_return:
4904                              <1> ;     retn
4905 00008A53 E9EDE5FFFF          <1>       jmp    print_msg
4906                              <1>
4907                              <1> cmp_cmd:
4908                              <1>        ; 29/01/2016 (TRDOS 386 = TRDOS v2.0)
4909 00008A58 BE[1E820100]        <1>         mov esi, CommandBuffer
4910                              <1>         ; edi = internal command word (ASCIIZ)
4911                              <1>        ; ecx = command length (<=8)
4912                              <1> cmp_cmd_1:
4913 00008A5D AC                  <1>       lodsb
4914 00008A5E AE                  <1>       scasb
4915 00008A5F 750D                <1>       jne    short cmp_cmd_3
4916 00008A61 E2FA                <1>       loop   cmp_cmd_1
4917 00008A63 AC                  <1>       lodsb
4918 00008A64 3C20                <1>       cmp    al, 20h
4919 00008A66 7703                <1>       ja     short cmp_cmd_2
4920 00008A68 30C0                <1>       xor    al, al
4921                              <1>       ; ZF = 1 -> internal command word matches
4922 00008A6A C3                  <1>       retn
4923                              <1> cmp_cmd_2:
4924                              <1>       ; ZF = 0 (CF = 0) -> external command word
4925 00008A6B 58                  <1>       pop    eax ; no return to the caller from here
4926 00008A6C EBCD                <1>       jmp    cmp_cmd_external
4927                              <1> cmp_cmd_3:
4928 00008A6E F9                  <1>       stc
4929                              <1>       ; CF = 1 -> internal command word does not match
4930 00008A6F C3                  <1>       retn
4931                              <1>
4932                              <1> loc_run_cmd_failed:
4933                              <1>       ; 15/03/2016
4934                              <1>       ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
4935                              <1>       ; 07/12/2009 (CMD_INTR.ASM)
4936                              <1>       ; 29/11/2009
4937                              <1>
4938 00008A70 E863000000          <1>       call   restore_cdir_after_cmd_fail
4939                              <1>
4940                              <1> loc_run_cmd_failed_cmp_al:
4941                              <1>       ; End of Restore_CDIR code (29/11/2009)
4942                              <1>
4943 00008A75 3C01                <1>       cmp    al, 1 ; Bad command or file name
4944 00008A77 74CC                <1>       je     loc_cmd_failed
4945                              <1> loc_run_dir_not_found:
4946 00008A79 3C03                <1>       cmp    al, 3
```

```
4947 00008A7B 750A              <1>       jne    short loc_run_file_notfound_msg
4948                            <1>       ; Path not found (MS-DOS Error Code = 3)
4949 00008A7D BE[573C0100]      <1>       mov    esi, Msg_Dir_Not_Found
4950 00008A82 E9BEE5FFFF        <1>       jmp    print_msg
4951                            <1>
4952                            <1> loc_run_file_notfound_msg:
4953 00008A87 3C02              <1>       cmp    al, 2 ; File not found
4954 00008A89 750A              <1>       jne    short loc_run_file_drv_read_err
4955                            <1>
4956                            <1> loc_print_file_notfound_msg:
4957 00008A8B BE[6E3C0100]      <1>         mov     esi, Msg_File_Not_Found
4958                            <1>       ;call  proc_printmsg
4959                            <1>       ;retn
4960 00008A90 E9B0E5FFFF        <1>       jmp    print_msg
4961                            <1>
4962                            <1> loc_run_file_drv_read_err:
4963                            <1>       ; Err: 17 (Read fault)
4964 00008A95 3C11              <1>       cmp    al, 17 ; Drive not ready or read error
4965 00008A97 7404              <1>       je     short loc_run_file_print_drv_read_err
4966                            <1>       ;
4967 00008A99 3C0F              <1>       cmp    al, 15 ; Drive not ready (or read error)
4968 00008A9B 750A              <1>       jne    short loc_run_file_toobig
4969                            <1>
4970                            <1> loc_run_file_print_drv_read_err:
4971 00008A9D BE[143C0100]      <1>       mov    esi, Msg_Not_Ready_Read_Err
4972 00008AA2 E99EE5FFFF        <1>       jmp    print_msg
4973                            <1>
4974                            <1> loc_run_file_toobig:
4975 00008AA7 3C08              <1>       cmp    al, 8 ; Not enough free memory to load&run file
4976 00008AA9 750A              <1>       jne    short loc_run_file_perm_denied
4977 00008AAB BE[B93C0100]      <1>       mov    esi, Msg_Insufficient_Memory
4978 00008AB0 E990E5FFFF        <1>       jmp    print_msg
4979                            <1>
4980                            <1> loc_run_file_perm_denied:
4981                            <1>       ; 29/12/2017
4982 00008AB5 3C0B              <1>       cmp    al, ERR_PERM_DENIED ; 11 ; Permission denied
4983 00008AB7 750A              <1>       jne    short loc_run_misc_error
4984 00008AB9 BE[4E3E0100]      <1>       mov    esi, Msg_Permission_Denied
4985 00008ABE E982E5FFFF        <1>       jmp    print_msg
4986                            <1>
4987                            <1>       ; 15/03/2016
4988                            <1> print_misc_error_msg:
4989                            <1> loc_run_misc_error:
4990                            <1>       ; AL = Error code
4991 00008AC3 E847B7FFFF        <1>       call   bytetohex
4992 00008AC8 66A3[ED3C0100]    <1>         mov     [error_code_hex], ax
4993                            <1>
4994 00008ACE BE[D03C0100]      <1>       mov    esi, Msg_Error_Code
4995                            <1>       ;call  print_msg
4996                            <1>       ;retn
4997                            <1>
4998 00008AD3 E96DE5FFFF        <1>       jmp    print_msg
4999                            <1>
5000                            <1> restore_cdir_after_cmd_fail:
5001                            <1>       ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
5002 00008AD8 50                <1>       push   eax
5003 00008AD9 8A3D[CA880100]    <1>       mov    bh, [RUN_CDRV] ; it is set at the beginning
5004                            <1>                               ; of the 'run' command.
5005 00008ADF 3A3D[6E810100]    <1>       cmp    bh, [Current_Drv]
5006 00008AE5 7409              <1>       je     short loc_run_restore_cdir
5007 00008AE7 88FA              <1>       mov    dl, bh
5008 00008AE9 E8C4F0FFFF        <1>       call   change_current_drive
5009 00008AEE EB19              <1>       jmp    short loc_run_err_pass_restore_cdir
5010                            <1>
5011                            <1> loc_run_restore_cdir:
5012 00008AF0 803D[55390100]00  <1>       cmp    byte [Restore_CDIR], 0
5013 00008AF7 7610              <1>       jna    short loc_run_err_pass_restore_cdir
5014 00008AF9 30DB              <1>       xor    bl, bl
5015 00008AFB 0FB7F3            <1>       movzx  esi, bx
5016 00008AFE 81C600010900      <1>       add    esi, Logical_DOSDisks
5017 00008B04 E860F1FFFF        <1>       call   restore_current_directory
5018                            <1>
5019                            <1> loc_run_err_pass_restore_cdir:
5020 00008B09 58                <1>       pop    eax
5021 00008B0A C3                <1>       retn
5022                            <1>
5023                            <1> print_directory_list:
5024                            <1>       ; 10/02/2016
5025                            <1>       ; 08/02/2016 (TRDOS 386 = TRDOS v2.0)
5026                            <1>       ; 06/12/2009 ('cmp_cmd_dir')
5027                            <1>       ;
5028 00008B0B 66C705[0C8A0100]00- <1>     mov    word [AttributesMask], 0800h ; ..except volume names..
5028 00008B13 08                <1>
5029 00008B14 A0[6E810100]      <1>       mov    al, [Current_Drv]
5030 00008B19 A2[CA880100]      <1>       mov    [RUN_CDRV], al
5031                            <1> get_dfname_fchar:
5032 00008B1E AC                <1>       lodsb
5033 00008B1F 3C20              <1>       cmp    al, 20h
5034 00008B21 74FB              <1>       je     short get_dfname_fchar
5035 00008B23 0F82A4000000      <1>         jb      loc_print_dir_call_all
5036 00008B29 3C2D              <1>       cmp    al, '-'
5037 00008B2B 7542              <1>       jne    short loc_print_dir_call_flt
5038                            <1> get_next_attr_char:
5039 00008B2D AC                <1>       lodsb
5040 00008B2E 3C20              <1>       cmp    al, 20h
5041 00008B30 74FB              <1>       je     short get_next_attr_char
5042 00008B32 0F820DFFFFFF      <1>         jb      loc_cmd_failed
5043 00008B38 24DF              <1>       and    al, 0DFh
5044 00008B3A 3C44              <1>       cmp    al, 'D' ; directories only ?
5045 00008B3C 7512              <1>       jne    short pass_only_directories
5046 00008B3E AC                <1>       lodsb
5047 00008B3F 3C20              <1>       cmp    al, 20h
5048 00008B41 0F87FEFEFFFF      <1>         ja      loc_cmd_failed
5049 00008B47 800D[0C8A0100]10  <1>       or     byte [AttributesMask], 10h ; ..directory..
5050 00008B4E EB18              <1>       jmp    short get_dfname_fchar_attr
```

```
                            <1> pass_only_directories:
00008B50 3C46               <1>       cmp    al, 'F'      ; files only ?
00008B52 0F85B0000000       <1>         jne     check_attr_s
00008B58 AC                 <1>       lodsb
00008B59 3C20               <1>       cmp    al, 20h
00008B5B 0F87E4FEFFFF       <1>         ja      loc_cmd_failed
00008B61 800D[0D8A0100]10   <1>       or     byte [AttributesMask+1], 10h ; ..except directories..
                            <1> get_dfname_fchar_attr:
00008B68 AC                 <1>       lodsb
00008B69 3C20               <1>       cmp    al, 20h
00008B6B 74FB               <1>       je     short get_dfname_fchar_attr
00008B6D 725E               <1>       jb     short loc_print_dir_call_all
                            <1>
                            <1> loc_print_dir_call_flt:
00008B6F 4E                 <1>       dec    esi
00008B70 BF[0E8A0100]       <1>       mov    edi, FindFile_Drv
00008B75 E8AC250000         <1>       call   parse_path_name
00008B7A 7308               <1>       jnc    short loc_print_dir_change_drv_1
00008B7C 3C01               <1>       cmp    al, 1
00008B7E 0F87ECFEFFFF       <1>         ja      loc_run_cmd_failed
                            <1>
                            <1> loc_print_dir_change_drv_1:
00008B84 8A15[0E8A0100]     <1>       mov    dl, [FindFile_Drv]
                            <1> loc_print_dir_change_drv_2:
00008B8A 3A15[CA880100]     <1>       cmp    dl, [RUN_CDRV]
00008B90 740B               <1>       je     short loc_print_dir_change_directory
00008B92 E81BF0FFFF         <1>       call   change_current_drive
00008B97 0F82D3FEFFFF       <1>         jc      loc_run_cmd_failed
                            <1> loc_print_dir_change_directory:
00008B9D 803D[0F8A0100]20   <1>       cmp    byte [FindFile_Directory], 20h ; 0 or 20h ?
00008BA4 761D               <1>       jna    short pass_print_dir_change_directory
                            <1>
00008BA6 FE05[55390100]     <1>       inc    byte [Restore_CDIR]
00008BAC BE[0F8A0100]       <1>       mov    esi, FindFile_Directory
00008BB1 30E4               <1>       xor    ah, ah ; CD_COMMAND sign -> 0
00008BB3 E8581F0000         <1>       call   change_current_directory
00008BB8 0F82B2FEFFFF       <1>         jc      loc_run_cmd_failed
                            <1>
                            <1> loc_print_dir_change_prompt_dir_string:
00008BBE E86D1E0000         <1>       call   change_prompt_dir_string
                            <1>
                            <1> pass_print_dir_change_directory:
00008BC3 BE[508A0100]       <1>       mov    esi, FindFile_Name
00008BC8 803E20             <1>       cmp    byte [esi], 20h ; ; 0 or 20h ?
00008BCB 7706               <1>       ja     short loc_print_dir_call
                            <1>
                            <1> loc_print_dir_call_all:
00008BCD C7062A2E2A00       <1>       mov    dword [esi], '*.*'
                            <1> loc_print_dir_call:
00008BD3 E87E000000         <1>       call   print_directory
                            <1>
00008BD8 8A15[CA880100]     <1>       mov    dl, [RUN_CDRV]  ; it is set at the beginning
00008BDE 3A15[6E810100]     <1>       cmp    dl, [Current_Drv]
00008BE4 7406               <1>       je     short loc_print_dir_call_restore_cdir_retn
00008BE6 E8C7EFFFFF         <1>       call   change_current_drive
00008BEB C3                 <1>       retn
                            <1>
                            <1> loc_print_dir_call_restore_cdir_retn:
00008BEC 803D[55390100]00   <1>       cmp    byte [Restore_CDIR], 0
00008BF3 7610               <1>       jna    short pass_print_dir_call_restore_cdir_retn
                            <1>
00008BF5 BE00010900         <1>       mov    esi, Logical_DOSDisks
00008BFA 31C0               <1>       xor    eax, eax
00008BFC 88D4               <1>       mov    ah, dl
00008BFE 01C6               <1>       add    esi, eax
                            <1>
00008C00 E864F0FFFF         <1>       call   restore_current_directory
                            <1>
                            <1> pass_print_dir_call_restore_cdir_retn:
00008C05 C3                 <1>       retn
                            <1>
                            <1> check_attr_s_cap:
00008C06 24DF               <1>       and    al, 0DFh
                            <1> check_attr_s:
00008C08 3C53               <1>       cmp    al, 'S'
00008C0A 7514               <1>       jne    short pass_attr_s
00008C0C 800D[0C8A0100]04   <1>       or     byte [AttributesMask], 4 ; system
00008C13 AC                 <1>       lodsb
00008C14 3C20               <1>       cmp    al, 20h
00008C16 0F844CFFFFFF       <1>         je      get_dfname_fchar_attr
00008C1C 72AF               <1>       jb     short loc_print_dir_call_all
00008C1E 24DF               <1>       and    al, 0DFh
                            <1> pass_attr_s:
00008C20 3C48               <1>       cmp    al, 'H'
00008C22 7514               <1>       jne    short pass_attr_h
00008C24 800D[0C8A0100]02   <1>       or     byte [AttributesMask], 2 ; hidden
                            <1> pass_attr_shr:
00008C2B AC                 <1>       lodsb
00008C2C 3C20               <1>       cmp    al, 20h
00008C2E 0F8434FFFFFF       <1>         je      get_dfname_fchar_attr
00008C34 7297               <1>       jb     short loc_print_dir_call_all
00008C36 EBCE               <1>       jmp    short check_attr_s_cap
                            <1>
                            <1> pass_attr_h:
00008C38 3C52               <1>       cmp    al, 'R'
00008C3A 7509               <1>       jne    short pass_attr_r
00008C3C 800D[0C8A0100]01   <1>       or     byte [AttributesMask], 1 ; read only
00008C43 EBE6               <1>       jmp    short pass_attr_shr
                            <1>
                            <1> pass_attr_r:
00008C45 3C41               <1>       cmp    al, 'A'
00008C47 0F85F8FDFFFF       <1>         jne     loc_cmd_failed
00008C4D 800D[0C8A0100]20   <1>       or     byte [AttributesMask], 20h ; archive
00008C54 EBD5               <1>       jmp    short pass_attr_shr
                            <1>
```

```
                        <1> print_directory:
                        <1>       ; 13/05/2016
                        <1>       ; 11/02/2016
                        <1>       ; 10/02/2016
                        <1>       ; 08/02/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>       ; 30/10/2010 ('proc_print_directory')
                        <1>       ; 19/09/2009
                        <1>       ; 2005
                        <1>       ; INPUT ->
                        <1>       ;      ESI = Asciiz File/Dir Name Address
                        <1>
00008C56 56             <1>       push   esi
                        <1>
00008C57 29C0           <1>       sub    eax, eax
                        <1>
00008C59 66A3[988A0100] <1>       mov    word [Dir_Count], ax ; 0
00008C5F 66A3[968A0100] <1>       mov    word [File_Count], ax ; 0
00008C65 A3[9A8A0100]   <1>       mov    dword [Total_FSize], eax ; 0
                        <1>
00008C6A E8ECE3FFFF     <1>       call    clear_screen
                        <1>
00008C6F 31C9           <1>       xor    ecx, ecx
00008C71 8A2D[6E810100] <1>       mov     ch, [Current_Drv] ; DirBuff_Drv - 'A'
00008C77 A0[6F810100]   <1>       mov     al, [Current_Dir_Drv]
00008C7C A2[123B0100]   <1>       mov     [Dir_Drive_Name], al
00008C81 BE00010900     <1>       mov    esi, Logical_DOSDisks
00008C86 01CE           <1>       add    esi, ecx
                        <1>
00008C88 E858F9FFFF     <1>       call   move_volume_name_and_serial_no
00008C8D 730C           <1>       jnc    short print_dir_strlen_check
                        <1>
00008C8F 5E             <1>       pop    esi
00008C90 8A3D[D6800100] <1>       mov    bh, [ptty] ; [ACTIVE_PAGE]
                        <1>       ;call  beeper
                        <1>       ;retn
00008C96 E9B196FFFF     <1>       jmp    beeper  ; beep ! and return
                        <1>
                        <1> print_dir_strlen_check:
00008C9B BE[71810100]   <1>       mov    esi, Current_Dir_Root
00008CA0 BF[AF3B0100]   <1>       mov    edi, Dir_Str_Root
                        <1>
                        <1>       ;xor   ecx, ecx
00008CA5 8A0D[CD810100] <1>         mov     cl, [Current_Dir_StrLen]
00008CAB FEC1           <1>       inc    cl
00008CAD 80F940         <1>       cmp    cl, 64
00008CB0 760D           <1>       jna    short pass_print_dir_strlen_shorting
00008CB2 46             <1>       inc    esi
00008CB3 01CE           <1>       add    esi, ecx
00008CB5 83EE40         <1>       sub    esi, 64
00008CB8 47             <1>       inc    edi
00008CB9 B82E2E2E20     <1>       mov    eax, '... '
00008CBE AB             <1>       stosd
                        <1>
                        <1> pass_print_dir_strlen_shorting:
00008CBF F3A4           <1>       rep    movsb
                        <1>
00008CC1 BE[053B0100]   <1>       mov    esi, Dir_Drive_Str
00008CC6 E87AE3FFFF     <1>       call   print_msg
                        <1>
00008CCB BE[643B0100]   <1>       mov    esi, Vol_Serial_Header
00008CD0 E870E3FFFF     <1>       call   print_msg
                        <1>
00008CD5 BE[A43B0100]   <1>       mov    esi, Dir_Str_Header
00008CDA E866E3FFFF     <1>       call   print_msg
                        <1>
00008CDF BE[D5430100]   <1>       mov    esi, next2line
00008CE4 E85CE3FFFF     <1>       call   print_msg
                        <1>
                        <1> loc_print_dir_first_file:
00008CE9 C605[AD8A0100]10 <1>     mov    byte [PrintDir_RowCounter], 16
00008CF0 66A1[0C8A0100] <1>       mov    ax, [AttributesMask]
00008CF6 5E             <1>       pop    esi
                        <1>
00008CF7 E859020000     <1>       call   find_first_file
00008CFC 0F826F010000   <1>         jc     loc_dir_ok
                        <1>
                        <1> loc_dfname_use_this:
                        <1>       ; bl = File Attributes (bh = Long Name Entry Length)
00008D02 F6C310         <1>       test   bl, 10h  ; Is it a directory?
00008D05 741B           <1>       jz     short loc_not_dir
                        <1>
00008D07 66FF05[988A0100] <1>     inc    word [Dir_Count]
00008D0E 89F2           <1>       mov    edx, esi      ; FindFile_DirEntry address
00008D10 BE[F43C0100]   <1>       mov    esi, Type_Dir; '<DIR>     '
00008D15 BF[0B3D0100]   <1>       mov    edi, Dir_Or_FileSize
                        <1>       ; move 10 bytes
00008D1A A5             <1>       movsd
00008D1B A5             <1>       movsd
00008D1C 66A5           <1>       movsw
00008D1E 89D6           <1>       mov    esi, edx
00008D20 EB36           <1>       jmp     short loc_dir_attribute
                        <1>
                        <1> loc_not_dir:
00008D22 66FF05[968A0100] <1>     inc    word [File_Count]
00008D29 0105[9A8A0100] <1>       add    [Total_FSize], eax
                        <1>
00008D2F B90A000000     <1>       mov    ecx, 10  ; 32 bit divisor
00008D34 89CF           <1>       mov    edi, ecx
00008D36 81C7[0B3D0100] <1>       add    edi, Dir_Or_FileSize
                        <1> loc_dir_rdivide:
00008D3C 29D2           <1>       sub    edx, edx
00008D3E F7F1           <1>       div    ecx     ; remainder in dl (< 10)
00008D40 80C230         <1>       add     dl, '0'      ; to make visible (ascii)
00008D43 4F             <1>       dec    edi
00008D44 8817           <1>       mov     [edi], dl
```

```
                                <1>       and    eax, eax
00008D48 75F2                   <1>       jnz    short loc_dir_rdivide
                                <1>
                                <1> loc_dir_fill_space:
00008D4A 81FF[0B3D0100]         <1>       cmp    edi, Dir_Or_FileSize
00008D50 7606                   <1>       jna    short loc_dir_attribute
00008D52 4F                     <1>       dec    edi
00008D53 C60720                 <1>       mov    byte [edi], 20h
00008D56 EBF2                   <1>       jmp    short loc_dir_fill_space
                                <1>
                                <1> loc_dir_attribute:
00008D58 C705[163D0100]2020-    <1>       mov    dword [File_Attribute], 20202020h
00008D60 2020                   <1>
                                <1>
00008D62 80FB20                 <1>       cmp    bl, 20h  ; Is it an archive file?
00008D65 7207                   <1>       jb     short loc_dir_pass_arch
00008D67 C605[193D0100]41       <1>       mov    byte [File_Attribute+3], 'A'
                                <1>
                                <1> loc_dir_pass_arch:
00008D6E 80E307                 <1>       and    bl, 7
00008D71 7428                   <1>       jz     short loc_dir_file_name
00008D73 88DF                   <1>       mov    bh, bl
00008D75 80E303                 <1>       and    bl, 3
00008D78 38DF                   <1>       cmp    bh, bl
00008D7A 7607                   <1>       jna    short loc_dir_pass_s
00008D7C C605[163D0100]53       <1>       mov    byte [File_Attribute], 'S'
                                <1>
                                <1> loc_dir_pass_s:
00008D83 80E302                 <1>       and    bl,2
00008D86 7407                   <1>       jz     short loc_dir_pass_h
00008D88 C605[173D0100]48       <1>       mov    byte [File_Attribute+1], 'H'
                                <1> loc_dir_pass_h:
00008D8F 80E701                 <1>       and    bh,1
00008D92 7407                   <1>       jz     short loc_dir_file_name
00008D94 C605[183D0100]52       <1>       mov    byte [File_Attribute+2], 'R'
                                <1> loc_dir_file_name:
                                <1>       ;mov   bx, [esi+18h] ; Date
                                <1>       ;mov   dx, [esi+16h] ; Time
00008D9B 8B5E16                 <1>       mov    ebx, [esi+16h]
00008D9E 89F1                   <1>       mov    ecx, esi ; FindFile_DirEntry address
00008DA0 BF[FE3C0100]           <1>       mov   edi, File_Name
                                <1>       ; move 8 bytes
00008DA5 A5                     <1>       movsd
00008DA6 A5                     <1>       movsd
00008DA7 C60720                 <1>       mov    byte [edi], 20h
00008DAA 47                     <1>       inc    edi
                                <1>       ; move 3 bytes
00008DAB 66A5                   <1>       movsw
00008DAD A4                     <1>       movsb
00008DAE 89CE                   <1>       mov    esi, ecx
                                <1>
                                <1> Dir_Time_start:
                                <1>       ;mov   ax, dx       ; Time
00008DB0 6689D8                 <1>       mov    ax, bx
00008DB3 66C1E805               <1>       shr    ax, 5        ; shift right 5 times
00008DB7 6683E03F               <1>       and    ax, 0000111111b     ; Minute Mask
00008DBB D40A                   <1>       aam                 ; Q([AL]/10)->AH
                                <1>                           ; R([AL]/10)->AL
                                <1>                           ; [AL]+[AH]= Minute as BCD
00008DBD 660D3030               <1>       or     ax, '00'     ; Convert to ASCII
00008DC1 86E0                   <1>       xchg   ah, al
00008DC3 66A3[293D0100]         <1>       mov    [File_Minute], ax
                                <1>
                                <1>       ;mov   al, dh
00008DC9 88F8                   <1>       mov    al, bh
00008DCB C0E803                 <1>       shr    al, 3        ; shift right 3 times
00008DCE D40A                   <1>       aam                 ; [AL]+[AH]= Hours as BCD
00008DD0 660D3030               <1>       or     ax, '00'
00008DD4 86E0                   <1>       xchg   ah, al
00008DD6 66A3[263D0100]         <1>       mov     [File_Hour], ax
                                <1>
00008DDC C1EB10                 <1>       shr    ebx, 16            ; BX = Date
                                <1>
                                <1> Dir_Date_start:
00008DDF 6689D8                 <1>       mov    ax, bx       ; Date
00008DE2 6683E01F               <1>       and    ax, 00011111b; Day Mask
00008DE6 D40A                   <1>       aam                 ; Q([AL]/10)->AH
                                <1>                           ; R([AL]/10)->AL
                                <1>                           ; [AL]+[AH]= Day as BCD
00008DE8 660D3030               <1>       or     ax, '00'     ; Convert to ASCII
00008DEC 86C4                   <1>       xchg   al, ah
                                <1>
00008DEE 66A3[1B3D0100]         <1>       mov    [File_Day], ax
                                <1>
00008DF4 6689D8                 <1>       mov    ax, bx
00008DF7 66C1E805               <1>       shr    ax, 5        ; shift right 5 times
00008DFB 6683E00F               <1>       and    ax, 00001111b; Month Mask
00008DFF D40A                   <1>       aam
00008E01 660D3030               <1>       or     ax, '00'
00008E05 86E0                   <1>       xchg   ah, al
00008E07 66A3[1E3D0100]         <1>       mov    [File_Month], ax
                                <1>
00008E0D 6689D8                 <1>       mov    ax, bx
00008E10 66C1E809               <1>       shr     ax, 9
00008E14 6683E07F               <1>       and    ax, 01111111b; Result = Year - 1980
00008E18 6605BC07               <1>       add    ax, 1980
                                <1>
00008E1C B10A                   <1>       mov    cl, 10
00008E1E F6F1                   <1>       div    cl           ; Q -> AL, R -> AH
00008E20 80CC30                 <1>       or     ah, '0'
00008E23 8825[243D0100]         <1>       mov    [File_Year+3], ah
00008E29 D40A                   <1>       aam
00008E2B 86E0                   <1>       xchg   ah, al
00008E2D 80CC30                 <1>       or     ah, '0'        ; Convert to ASCII
00008E30 8825[233D0100]         <1>       mov    [File_Year+2], ah
```

```
5365 00008E36 D40A               <1>       aam
5366 00008E38 86C4               <1>       xchg   al, ah
5367 00008E3A 660D3030           <1>       or     ax, '00'
5368 00008E3E 66A3[213D0100]     <1>       mov    [File_Year], ax
5369                             <1>
5370                             <1> loc_show_line:
5371 00008E44 56                 <1>       push   esi
5372 00008E45 BE[FE3C0100]       <1>       mov   esi, File_Name
5373 00008E4A E8F6E1FFFF         <1>       call   print_msg
5374 00008E4F BE[D7430100]       <1>       mov    esi, nextline
5375 00008E54 E8ECE1FFFF         <1>       call   print_msg
5376 00008E59 5E                 <1>       pop    esi
5377                             <1>
5378 00008E5A FE0D[AD8A0100]     <1>       dec    byte [PrintDir_RowCounter]
5379 00008E60 0F84D4000000       <1>       jz    pause_dir_scroll
5380                             <1>
5381                             <1> loc_next_entry:
5382 00008E66 E899010000         <1>       call   find_next_file
5383 00008E6B 0F8391FEFFFF       <1>       jnc   loc_dfname_use_this
5384                             <1>
5385                             <1> loc_dir_ok:
5386 00008E71 B90A000000         <1>       mov   ecx, 10
5387 00008E76 66A1[988A0100]     <1>       mov    ax, [Dir_Count]
5388 00008E7C BF[3F3D0100]       <1>       mov    edi, Decimal_Dir_Count
5389 00008E81 6639C8             <1>       cmp    ax, cx ; 10
5390 00008E84 7216               <1>       jb     short pass_ddc
5391 00008E86 47                 <1>       inc    edi
5392 00008E87 6683F864           <1>       cmp    ax, 100
5393 00008E8B 720F               <1>       jb     short pass_ddc
5394 00008E8D 47                 <1>       inc    edi
5395 00008E8E 663DE803           <1>       cmp    ax, 1000
5396 00008E92 7208               <1>       jb     short pass_ddc
5397 00008E94 47                 <1>       inc    edi
5398 00008E95 663D1027           <1>       cmp    ax, 10000
5399 00008E99 7201               <1>       jb     short pass_ddc
5400 00008E9B 47                 <1>       inc    edi
5401                             <1> pass_ddc:
5402 00008E9C 886F01             <1>       mov     [edi+1], ch ; 0
5403                             <1> loc_ddc_rediv:
5404 00008E9F 31D2               <1>       xor     edx, edx
5405 00008EA1 66F7F1             <1>       div     cx   ; 10
5406 00008EA4 80C230             <1>       add     dl, '0'
5407 00008EA7 8817               <1>       mov     [edi], dl
5408 00008EA9 4F                 <1>       dec     edi
5409 00008EAA 6609C0             <1>       or     ax, ax
5410 00008EAD 75F0               <1>       jnz    short loc_ddc_rediv
5411                             <1>
5412 00008EAF 66A1[968A0100]     <1>       mov     ax, [File_Count]
5413 00008EB5 BF[2E3D0100]       <1>       mov     edi, Decimal_File_Count
5414 00008EBA 6639C8             <1>       cmp     ax, cx ; 10
5415 00008EBD 7216               <1>       jb      short pass_dfc
5416 00008EBF 47                 <1>       inc     edi
5417 00008EC0 6683F864           <1>       cmp     ax, 100
5418 00008EC4 720F               <1>       jb      short pass_dfc
5419 00008EC6 47                 <1>       inc     edi
5420 00008EC7 663DE803           <1>       cmp     ax, 1000
5421 00008ECB 7208               <1>       jb      short pass_dfc
5422 00008ECD 47                 <1>       inc     edi
5423 00008ECE 663D1027           <1>       cmp     ax, 10000
5424 00008ED2 7201               <1>       jb      short pass_dfc
5425 00008ED4 47                 <1>       inc     edi
5426                             <1> pass_dfc:
5427                             <1>       ;mov    cx, 10
5428 00008ED5 886F01             <1>       mov     [edi+1], ch ; 00
5429                             <1> loc_dfc_rediv:
5430                             <1>       ;xor   dx, dx
5431 00008ED8 30D2               <1>       xor    dl, dl
5432 00008EDA 66F7F1             <1>       div    cx
5433 00008EDD 80C230             <1>       add    dl, '0'
5434 00008EE0 8817               <1>       mov    [edi], dl
5435 00008EE2 4F                 <1>       dec    edi
5436 00008EE3 6609C0             <1>       or     ax, ax
5437 00008EE6 75F0               <1>       jnz    short loc_dfc_rediv
5438                             <1>
5439 00008EE8 BF[AC8A0100]       <1>       mov     edi, TFS_Dec_End
5440                             <1>       ;mov    byte [edi], 0
5441 00008EED A1[9A8A0100]       <1>       mov     eax, [Total_FSize]
5442                             <1>       ;mov    ecx, 10
5443                             <1> rediv_tfs_hex:
5444                             <1>       ;sub   edx, edx
5445 00008EF2 28D2               <1>       sub    dl, dl
5446 00008EF4 F7F1               <1>       div    ecx
5447 00008EF6 80C230             <1>       add    dl, '0'
5448 00008EF9 4F                 <1>       dec   edi
5449 00008EFA 8817               <1>       mov   [edi], dl
5450 00008EFC 21C0               <1>       and    eax, eax
5451 00008EFE 75F2               <1>       jnz    short rediv_tfs_hex
5452                             <1>
5453 00008F00 893D[9E8A0100]     <1>       mov    [TFS_Dec_Begin], edi
5454 00008F06 BE[2C3D0100]       <1>       mov    esi, Decimal_File_Count_Header
5455 00008F0B E835E1FFFF         <1>       call   print_msg
5456 00008F10 BE[343D0100]       <1>       mov    esi, str_files
5457 00008F15 E82BE1FFFF         <1>       call   print_msg
5458 00008F1A BE[453D0100]       <1>       mov    esi, str_dirs
5459 00008F1F E821E1FFFF         <1>       call   print_msg
5460 00008F24 8B35[9E8A0100]     <1>       mov    esi, [TFS_Dec_Begin]
5461 00008F2A E816E1FFFF         <1>       call   print_msg
5462 00008F2F BE[563D0100]       <1>       mov    esi, str_bytes
5463 00008F34 E80CE1FFFF         <1>       call   print_msg
5464                             <1>
5465 00008F39 C3                 <1>       retn
5466                             <1>
5467                             <1> pause_dir_scroll:
5468 00008F3A 28E4               <1>       sub    ah, ah
5469 00008F3C E89A7FFFFF         <1>       call   int16h
```

```
5470 00008F41 3C1B                <1>       cmp    al, 1Bh
5471 00008F43 0F8428FFFFFF        <1>       je     loc_dir_ok
5472 00008F49 C605[AD8A0100]10    <1>       mov    byte [PrintDir_RowCounter], 16 ; Reset counter
5473 00008F50 E911FFFFFF          <1>       jmp    loc_next_entry
5474                              <1>
5475                              <1> find_first_file:
5476                              <1>       ; 11/02/2016
5477                              <1>       ; 10/02/2016
5478                              <1>       ; 08/02/2016 (TRDOS 386 = TRDOS v2.0)
5479                              <1>       ; 09/10/2011
5480                              <1>       ; 17/09/2009
5481                              <1>       ; 2005
5482                              <1>       ; INPUT ->
5483                              <1>       ;      ESI = ASCIIZ File/Dir Name Address (in Current Directory)
5484                              <1>       ;      AL = Attributes AND mask (The AND result must be equal to AL)
5485                              <1>       ;            bit 0 = Read Only
5486                              <1>       ;            bir 1 = Hidden
5487                              <1>       ;            bit 2 = System
5488                              <1>       ;            bit 3 = Volume Label
5489                              <1>       ;            bit 4 = Directory
5490                              <1>       ;            bit 5 = Archive
5491                              <1>       ;            bit 6 = Reserved, must be 0
5492                              <1>       ;            bit 7 = Reserved, must be 0
5493                              <1>       ;       AH = Attributes Negative AND mask (The AND result must be ZERO)
5494                              <1>       ;
5495                              <1>       ; OUTPUT ->
5496                              <1>       ;      CF = 1 -> Error, Error Code in EAX (AL)
5497                              <1>       ;      CF = 0 ->
5498                              <1>       ;           ESI = Directory Entry (FindFile_DirEntry) Location
5499                              <1>       ;           EDI = Directory Buffer Directory Entry Location
5500                              <1>       ;           EAX = File Size
5501                              <1>       ;            BL = Attributes of The File/Directory
5502                              <1>       ;            BH = Long Name Yes/No Status (>0 is YES)
5503                              <1>       ;            DX > 0 : Ambiguous filename chars are used
5504                              <1>       ;
5505                              <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
5506                              <1>
5507 00008F55 66A3[5E8A0100]      <1>       mov    [FindFile_AttributesMask], ax
5508 00008F5B BF[608A0100]        <1>       mov    edi, FindFile_DirEntry ; TR-DOS Fullfilename formatted buffer
5509 00008F60 31C0                <1>       xor    eax, eax
5510 00008F62 B90B000000          <1>       mov    ecx, 11
5511 00008F67 F3AB                <1>       rep    stosd  ; 44 bytes
5512                              <1>       ;stosw       ; +2 bytes
5513                              <1>
5514 00008F69 BF[508A0100]        <1>       mov    edi, FindFile_Name ; FFF structure, offset 66
5515 00008F6E 39FE                <1>       cmp    esi, edi
5516 00008F70 7408                <1>       je     short loc_fff_mfn_ok
5517 00008F72 89FA                <1>       mov    edx, edi
5518                              <1>       ; move 13 bytes
5519 00008F74 A5                  <1>       movsd
5520 00008F75 A5                  <1>       movsd
5521 00008F76 A5                  <1>       movsd
5522 00008F77 AA                  <1>       stosb
5523 00008F78 89D6                <1>       mov    esi, edx
5524                              <1> loc_fff_mfn_ok:
5525 00008F7A BF[FF890100]        <1>       mov    edi, Dir_Entry_Name ; Dir Entry Format File Name
5526 00008F7F E8D7200000          <1>       call   convert_file_name
5527 00008F84 89FE                <1>       mov    esi, edi ; offset Dir_Entry_Name
5528                              <1>
5529 00008F86 66A1[5E8A0100]      <1>       mov    ax, [FindFile_AttributesMask]
5530                              <1>       ;xor   ecx, ecx
5531 00008F8C 30C9                <1>       xor    cl, cl
5532 00008F8E E8D01D0000          <1>       call   locate_current_dir_file
5533 00008F93 726E                <1>       jc     short loc_fff_retn
5534                              <1>       ; EDI = Directory Entry
5535                              <1>       ; EBX = Directory Buffer Entry Index/Number
5536                              <1>
5537                              <1> loc_fff_fnf_ln_check:
5538 00008F95 30ED                <1>       xor    ch, ch
5539 00008F97 80F60F              <1>       xor    dh, 0Fh
5540 00008F9A 7408                <1>       jz     short loc_fff_longname_yes
5541 00008F9C 882D[5D8A0100]      <1>       mov    [FindFile_LongNameYes], ch ; 0
5542 00008FA2 EB0C                <1>       jmp    short loc_fff_longname_no
5543                              <1>
5544                              <1> loc_fff_longname_yes:
5545                              <1>       ;inc   byte [FindFile_LongNameYes]
5546 00008FA4 8A0D[6A890100]      <1>       mov    cl, [LFN_EntryLength]
5547 00008FAA 880D[5D8A0100]      <1>       mov    [FindFile_LongNameEntryLength], cl ; FindFile_LongNameYes
5548                              <1>
5549                              <1> loc_fff_longname_no:
5550                              <1>       ;mov   bx, [DirBuff_CurrentEntry]
5551 00008FB0 66891D[888A0100]    <1>       mov    [FindFile_DirEntryNumber], bx
5552 00008FB7 6689C2              <1>       mov    dx, ax ; Ambiguous Filename chars used sign > 0
5553                              <1>
5554 00008FBA A0[6E810100]        <1>       mov    al, [Current_Drv]
5555 00008FBF A2[0E8A0100]        <1>       mov    [FindFile_Drv], al
5556                              <1>
5557 00008FC4 A1[68810100]        <1>       mov    eax, [Current_Dir_FCluster]
5558 00008FC9 A3[808A0100]        <1>       mov    [FindFile_DirFirstCluster], eax
5559                              <1>
5560 00008FCE A1[99880100]        <1>       mov    eax, [DirBuff_Cluster]
5561 00008FD3 A3[848A0100]        <1>       mov    [FindFile_DirCluster], eax
5562                              <1>
5563 00008FD8 66FF05[8A8A0100]    <1>       inc    word [FindFile_MatchCounter]
5564                              <1>
5565 00008FDF 89FB                <1>       mov    ebx, edi
5566 00008FE1 89FE                <1>       mov    esi, edi
5567 00008FE3 BF[608A0100]        <1>       mov    edi, FindFile_DirEntry
5568 00008FE8 89F8                <1>       mov    eax, edi
5569 00008FEA B108                <1>       mov    cl, 8
5570 00008FEC F3A5                <1>       rep    movsd
5571 00008FEE 89C6                <1>       mov    esi, eax
5572 00008FF0 89DF                <1>       mov    edi, ebx
5573                              <1>
5574 00008FF2 A1[7C8A0100]        <1>       mov    eax, [FindFile_DirEntry+28] ; File Size
```

```
                                <1>
00008FF7 8A1D[6B8A0100]         <1>       mov   bl, [FindFile_DirEntry+11] ; File Attributes
00008FFD 8A3D[5D8A0100]         <1>       mov   bh, [FindFile_LongNameYes]
                                <1>
                                <1>       ;mov  cx, [DirBuff_EntryCounter]
                                <1>       ;mov  [FindFile_DirEntryNumber], cx
                                <1>       ;mov  cx, [FindFile_DirEntryNumber]
                                <1>       ; ecx = 0
                                <1>
                                <1> loc_fff_retn:
00009003 C3                     <1>       retn
                                <1>
                                <1> find_next_file:
                                <1>       ; 15/10/2016
                                <1>       ; 10/02/2016
                                <1>       ; 08/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 06/02/2011
                                <1>       ; 17/09/2009
                                <1>       ; 2005
                                <1>       ; INPUT ->
                                <1>       ;     NONE, Find First File Parameters
                                <1>       ; OUTPUT ->
                                <1>       ;     CF = 1 -> Error, Error Code in EAX (AL)
                                <1>       ;     CF = 0 ->
                                <1>       ;         ESI = Directory Entry (FindFile_DirEntry) Location
                                <1>       ;         EDI = Directory Buffer Directory Entry Location
                                <1>       ;         EAX = File Size
                                <1>       ;           BL = Attributes of The File/Directory
                                <1>       ;           BH = Long Name Yes/No Status (>0 is YES)
                                <1>       ;             DX > 0 : Ambiguous filename chars are used
                                <1>       ;
                                <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
                                <1>
00009004 66833D[8A8A0100]00     <1>       cmp   word [FindFile_MatchCounter], 0
0000900C 7707                   <1>       ja    short loc_start_search_next_file
                                <1>
                                <1> loc_fnf_stc_retn:
0000900E F9                     <1>       stc
                                <1> loc_fnf_ax12h_retn:
0000900F B80C000000             <1>       mov   eax, 12 ; No More files
                                <1> ;loc_fnf_retn:
00009014 C3                     <1>       retn
                                <1>
                                <1> loc_start_search_next_file:
00009015 668B1D[888A0100]       <1>       mov   bx, [FindFile_DirEntryNumber]
0000901C 6643                   <1>       inc   bx
0000901E 663B1D[97880100]       <1>       cmp   bx, [DirBuff_LastEntry]
00009025 7719                   <1>       ja    short loc_cont_search_next_file
                                <1>
                                <1> loc_fnf_search:
00009027 BE[FF890100]           <1>       mov   esi, Dir_Entry_Name
0000902C 66A1[5E8A0100]         <1>       mov   ax, [FindFile_AttributesMask]
00009032 6631C9                 <1>       xor   cx, cx
00009035 E82D1E0000             <1>       call  find_directory_entry
0000903A 0F8355FFFFFF           <1>         jnc     loc_fff_fnf_ln_check
                                <1>
                                <1> loc_cont_search_next_file:
00009040 31DB                   <1>       xor   ebx, ebx
00009042 8A3D[6E810100]         <1>       mov   bh, [Current_Drv]
00009048 BE00010900             <1>       mov   esi, Logical_DOSDisks
0000904D 01DE                   <1>       add   esi, ebx
                                <1>
0000904F 803D[6C810100]00       <1>       cmp   byte [Current_Dir_Level], 0
00009056 7608                   <1>       jna   short loc_fnf_check_FAT_type
00009058 807E0301               <1>       cmp   byte [esi+LD_FATType], 1
0000905C 72B1                   <1>       jb    short loc_fnf_ax12h_retn
0000905E EB06                   <1>       jmp   short loc_fnf_check_next_cluster
                                <1>
                                <1> loc_fnf_check_FAT_type:
00009060 807E0303               <1>       cmp   byte [esi+LD_FATType], 3
00009064 72A9                   <1>       jb    short loc_fnf_ax12h_retn
                                <1>
                                <1> loc_fnf_check_next_cluster:
00009066 A1[99880100]           <1>       mov   eax, [DirBuff_Cluster]
0000906B E8CA370000             <1>       call  get_next_cluster
00009070 7306                   <1>       jnc   short loc_fnf_load_next_dir_cluster
00009072 09C0                   <1>       or    eax, eax
00009074 7498                   <1>       jz    short loc_fnf_stc_retn
                                <1>       ;mov  eax, 17 ;Drive not ready or read error
00009076 F5                     <1>       cmc   ;stc
                                <1> loc_fnf_retn:
00009077 C3                     <1>       retn
                                <1>
                                <1> loc_fnf_load_next_dir_cluster:
00009078 E8A3390000             <1>       call  load_FAT_sub_directory
0000907D 72F8                   <1>       jc    short loc_fnf_retn
0000907F 6631DB                 <1>       xor   bx, bx
00009082 66891D[888A0100]       <1>       mov   [FindFile_DirEntryNumber], bx
00009089 EB9C                   <1>       jmp   short loc_fnf_search
                                <1>
                                <1> get_and_print_longname:
                                <1>       ; 16/10/2016
                                <1>       ; 13/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 24/01/2010
                                <1>       ; 17/10/2009 (CMD_INTR.ASM, 'cmp_cmd_longname')
                                <1> get_longname_fchar:
0000908B 803E20                 <1>       cmp   byte [esi], 20h
0000908E 7701                   <1>       ja    short loc_find_longname
                                <1>       ;jb   short loc_longname_retn
                                <1>       ;inc  esi
                                <1>       ;je   short get_longname_fchar
                                <1> ;loc_longname_retn:
00009090 C3                     <1>       retn
                                <1> loc_find_longname:
00009091 E839210000             <1>       call  find_longname
```

```
5680 00009096 7328              <1>       jnc    short loc_print_longname
5681                            <1>
5682 00009098 08C0              <1>       or     al, al
5683 0000909A 741A              <1>       jz     short loc_longname_not_found
5684                            <1>
5685                            <1>       ; 16/10/2016 (15h -> 15, 17)
5686 0000909C 3C0F              <1>       cmp    al, 15
5687 0000909E 0F84B6F7FFFF      <1>       je     cd_drive_not_ready ; drive not ready
5688                            <1>                                 ; or
5689 000090A4 3C11              <1>       cmp    al, 17              ; read error
5690 000090A6 0F84AEF7FFFF      <1>       je     cd_drive_not_ready
5691                            <1>
5692                            <1> loc_ln_file_dir_not_found:
5693 000090AC BE[803C0100]      <1>       mov    esi, Msg_File_Directory_Not_Found
5694                            <1>       ;call  print_msg
5695                            <1>         ;retn
5696 000090B1 E98FDFFFFF        <1>       jmp    print_msg
5697                            <1>
5698                            <1> loc_longname_not_found:
5699 000090B6 BE[9F3C0100]      <1>         mov     esi, Msg_LongName_Not_Found
5700                            <1>       ;call  print_msg
5701                            <1>         ;retn
5702 000090BB E985DFFFFF        <1>       jmp    print_msg
5703                            <1>
5704                            <1> loc_print_longname:
5705                            <1>       ;mov   esi, LongFileName
5706 000090C0 BF[6E820100]      <1>       mov    edi, TextBuffer
5707 000090C5 57                <1>       push   edi
5708 000090C6 3C00              <1>       cmp    al, 0
5709 000090C8 7708              <1>       ja     short loc_print_longname_1
5710                            <1> loc_print_FS_longname: ; Singlix FS (64 byte ASCIIZ file name)
5711 000090CA AC                <1>       lodsb
5712 000090CB AA                <1>       stosb
5713 000090CC 08C0              <1>       or     al, al
5714 000090CE 75FA              <1>       jnz    short loc_print_FS_longname
5715 000090D0 EB07              <1>       jmp    short loc_print_longname_2
5716                            <1>       ;
5717                            <1> loc_print_longname_1: ; MS Windows long name (UNICODE chars)
5718 000090D2 66AD              <1>       lodsw
5719 000090D4 AA                <1>       stosb
5720 000090D5 08C0              <1>       or     al, al
5721 000090D7 75F9              <1>       jnz    short loc_print_longname_1
5722                            <1>       ;
5723                            <1> loc_print_longname_2:
5724 000090D9 5E                <1>       pop    esi
5725 000090DA E866DFFFFF        <1>       call   print_msg
5726 000090DF BE[D7430100]      <1>       mov    esi, nextline
5727                            <1>       ;call  print_msg
5728                            <1>       ;retn
5729 000090E4 E95CDFFFFF        <1>       jmp    print_msg
5730                            <1>
5731                            <1> show_file:
5732                            <1>       ; 18/02/2016
5733                            <1>       ; 17/02/2016
5734                            <1>       ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
5735                            <1>       ; 13/09/2011 (CMD_INTR.ASM, 'cmp_cmd_show')
5736                            <1>       ; 08/11/2009
5737                            <1>
5738                            <1> loc_show_parse_path_name:
5739 000090E9 BF[0E8A0100]      <1>       mov    edi, FindFile_Drv
5740 000090EE E833200000        <1>       call   parse_path_name
5741 000090F3 0F824CF9FFFF      <1>       jc     loc_cmd_failed
5742                            <1>
5743                            <1> loc_show_check_filename_exists:
5744 000090F9 BE[508A0100]      <1>       mov    esi, FindFile_Name
5745 000090FE 803E20            <1>       cmp    byte [esi], 20h
5746 00009101 0F863EF9FFFF      <1>       jna    loc_cmd_failed
5747                            <1>
5748                            <1>       ; 15/02/2016 (invalid file name check)
5749 00009107 E807020000        <1>       call   check_filename
5750 0000910C 730A              <1>       jnc    short loc_show_change_drv
5751                            <1>
5752 0000910E BE[6C3D0100]      <1>       mov    esi, Msg_invalid_name_chars
5753 00009113 E92DDFFFFF        <1>       jmp    print_msg
5754                            <1>
5755                            <1> loc_show_change_drv:
5756 00009118 8A35[6E810100]    <1>       mov    dh, [Current_Drv]
5757 0000911E 8835[CA880100]    <1>       mov    [RUN_CDRV], dh
5758 00009124 8A15[0E8A0100]    <1>       mov    dl, [FindFile_Drv]
5759 0000912A 38F2              <1>       cmp    dl, dh
5760 0000912C 740B              <1>       je     short loc_show_change_directory
5761 0000912E E87FEAFFFF        <1>       call   change_current_drive
5762                            <1>       ;jc    loc_file_rw_cmd_failed
5763 00009133 0F8237F9FFFF      <1>       jc     loc_run_cmd_failed
5764                            <1>
5765                            <1> loc_show_change_directory:
5766 00009139 803D[0F8A0100]20  <1>       cmp    byte [FindFile_Directory], 20h
5767 00009140 7618              <1>       jna    short loc_findload_showfile
5768                            <1>
5769 00009142 FE05[55390100]    <1>       inc    byte [Restore_CDIR]
5770 00009148 BE[0F8A0100]      <1>       mov    esi, FindFile_Directory
5771 0000914D 30E4              <1>       xor    ah, ah ; CD_COMMAND sign -> 0
5772 0000914F E8BC190000        <1>       call   change_current_directory
5773                            <1>       ;jc    loc_file_rw_cmd_failed
5774 00009154 0F8216F9FFFF      <1>       jc     loc_run_cmd_failed
5775                            <1>
5776                            <1> ;loc_show_change_prompt_dir_string:
5777                            <1>       ;call  change_prompt_dir_string
5778                            <1>
5779                            <1> loc_findload_showfile:
5780                            <1>       ; 15/02/2016
5781 0000915A BE[508A0100]      <1>       mov    esi, FindFile_Name
5782 0000915F BF[FF890100]      <1>       mov    edi, Dir_Entry_Name ; Dir Entry Format File Name
5783 00009164 E8F21E0000        <1>       call   convert_file_name
5784 00009169 89FE              <1>       mov    esi, edi ; offset Dir_Entry_Name
```

```
                                <1>
0000916B 28C0                   <1>       sub    al, al ; Attrib AND mask = 0
                                <1>       ; Directory attribute : 10h
                                <1>       ; Volume name attribute: 8h
0000916D B418                   <1>       mov    ah, 00011000b ; 18h (Attrib NAND, AND --> zero mask)
                                <1>       ;
0000916F 6631C9                 <1>       xor    cx, cx
00009172 E8EC1B0000             <1>       call   locate_current_dir_file
                                <1>       ;jc    loc_file_rw_cmd_failed
00009177 0F82F3F8FFFF           <1>       jc     loc_run_cmd_failed
                                <1>
                                <1> loc_show_load_file:
                                <1>       ; EDI = Directory Entry
0000917D 668B4714               <1>       mov    ax, [edi+DirEntry_FstClusHI] ; First Cluster High Word
00009181 C1E010                 <1>       shl    eax, 16
00009184 668B471A               <1>       mov    ax, [edi+DirEntry_FstClusLO] ; First Cluster Low Word
00009188 A3[B88A0100]           <1>       mov    [Show_Cluster], eax
0000918D 8B471C                 <1>       mov    eax, [edi+DirEntry_FileSize] ; File Size
00009190 21C0                   <1>       and    eax, eax ; Empty file !
00009192 0F8491000000           <1>         jz     end_of_show_file
00009198 A3[BC8A0100]           <1>       mov    [Show_FileSize], eax
0000919D 31C0                   <1>       xor    eax, eax
0000919F A3[C08A0100]           <1>       mov    [Show_FilePointer], eax ; 0
000091A4 66A3[C48A0100]         <1>       mov    [Show_ClusterPointer], ax ; 0
000091AA 29DB                   <1>       sub    ebx, ebx
000091AC 8A3D[6E810100]         <1>       mov    bh, [Current_Drv]
000091B2 BE00010900             <1>       mov    esi, Logical_DOSDisks
000091B7 01DE                   <1>       add    esi, ebx
000091B9 8935[B48A0100]         <1>       mov    [Show_LDDDT], esi ; Logical DOS Drv Description Table addr
                                <1>
000091BF 807E0300               <1>       cmp    byte [esi+LD_FATType], 0
000091C3 7713                   <1>       ja     short loc_show_calculate_cluster_size
                                <1>       ; Singlix FS
                                <1>       ; First Cluster Number is FDT number (in compatibility buffer)
000091C5 8B15[B88A0100]         <1>       mov    edx, [Show_Cluster] ; Compatibility dir. buffer value (FDT)
000091CB 8915[B08A0100]         <1>       mov    [Show_FDT], edx
000091D1 31C0                   <1>       xor    eax, eax
000091D3 A3[B88A0100]           <1>       mov    [Show_Cluster], eax ; Sector index  = 0
                                <1>                                  ; (next time it will be 1)
                                <1> loc_show_calculate_cluster_size:
000091D8 668B5E11               <1>       mov    bx, [esi+LD_BPB+BPB_BytsPerSec] ; FAT 12-16-32 (512)
                                <1>       ; BX = 512 = [esi+LD_FS_BytesPerSec] ; Singlix FS
000091DC 8A4613                 <1>       mov    al, [esi+LD_BPB+BPB_SecPerClust] ; FAT 12-16-32 (<= 128)
                                <1>       ; AL = 1 = [esi+LD_FS_Reserved2] ; SectPerClust for Singlix FS
000091DF F7E3                   <1>       mul    ebx
                                <1>
                                <1>       ;cmp   eax, 65536 ; non-compatible (very big) cluster size
                                <1>       ;ja    short end_of_show_file
000091E1 66A3[C68A0100]         <1>       mov    [Show_ClusterSize], ax
                                <1>
                                <1> loc_start_show_file:
000091E7 BE[D7430100]           <1>       mov    esi, nextline
000091EC E854DEFFFF             <1>       call   print_msg
                                <1>
000091F1 A1[B88A0100]           <1>       mov    eax, [Show_Cluster]
000091F6 C605[C88A0100]17       <1>       mov    byte [Show_RowCount], 23
                                <1>
                                <1>       ; 17/02/2016
000091FD 8B35[B48A0100]         <1>       mov    esi, [Show_LDDDT]
                                <1>
                                <1> loc_show_next_cluster:
                                <1>       ; 15/02/2016
00009203 BB00000700             <1>       mov    ebx, Cluster_Buffer ; 70000h (for current TRDOS 386 version)
                                <1>       ; ESI = Logical DOS drv description table address
00009208 E851380000             <1>       call   read_cluster
                                <1>       ;jc    loc_file_rw_cmd_failed
0000920D 0F825DF8FFFF           <1>       jc     loc_run_cmd_failed
                                <1>
00009213 31DB                   <1>       xor    ebx, ebx
                                <1> loc_show_next_byte:
00009215 803D[C88A0100]00       <1>       cmp    byte [Show_RowCount], 0
0000921C 7521                   <1>       jne    short pass_show_wait_for_key
0000921E 30E4                   <1>       xor    ah, ah
00009220 E8B67CFFFF             <1>       call   int16h
00009225 3C1B                   <1>       cmp    al, 1Bh
00009227 750F                   <1>       jne    short pass_exit_show
                                <1> end_of_show_file:
                                <1> pass_show_file:
00009229 BE[D7430100]           <1>       mov    esi, nextline
0000922E E812DEFFFF             <1>       call   print_msg
00009233 E94B010000             <1>       jmp    loc_file_rw_restore_retn
                                <1>
                                <1> pass_exit_show:
00009238 C605[C88A0100]14       <1>       mov    byte [Show_RowCount], 20
                                <1> pass_show_wait_for_key:
0000923F 81C300000700           <1>       add    ebx, Cluster_Buffer
00009245 8A03                   <1>       mov    al, [ebx]
00009247 3C0D                   <1>       cmp    al, 0Dh
00009249 0F8590000000           <1>         jne    loc_show_check_tab_space
0000924F FE0D[C88A0100]         <1>       dec    byte [Show_RowCount]
                                <1> pass_show_dec_rowcount:
00009255 B307                   <1>       mov    bl, 7 ; (light gray character color, black background)
00009257 8A3D[D6800100]         <1>       mov    bh, [ACTIVE_PAGE] ; [ptty]
0000925D E80190FFFF             <1>       call   _write_tty
                                <1> loc_show_check_eof:
00009262 FF05[C08A0100]         <1>       inc    dword [Show_FilePointer]
00009268 A1[C08A0100]           <1>       mov    eax, [Show_FilePointer]
0000926D 3B05[BC8A0100]         <1>       cmp    eax, [Show_FileSize]
00009273 73B4                   <1>       jnb    short end_of_show_file
00009275 66FF05[C48A0100]       <1>       inc    word [Show_ClusterPointer]
0000927C 0FB71D[C48A0100]       <1>       movzx  ebx, word [Show_ClusterPointer]
                                <1>
                                <1>       ; 17/02/2016
                                <1>       ; (sector boundary -9 bits- check, 512 = 0)
00009283 66F7C3FF01             <1>         test    bx, 1FFh ;  1 to 511
```

```
5890 00009288 758B                <1>       jnz    short loc_show_next_byte
5891                              <1>
5892                              <1>       ; 16/02/2016
5893 0000928A 8B35[B48A0100]      <1>       mov    esi, [Show_LDDDT]
5894                              <1>       ;
5895 00009290 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
5896 00009294 7719                <1>       ja     short loc_show_check_fat_cluster_size
5897                              <1>
5898                              <1>       ; Singlix FS
5899                              <1>       ; 1 sector, more... (cluster size = 1 sector)
5900 00009296 A1[B88A0100]        <1>       mov    eax, [Show_Cluster]
5901 0000929B 40                  <1>       inc    eax
5902 0000929C A3[B88A0100]        <1>       mov    [Show_Cluster], eax
5903                              <1>
5904 000092A1 6621DB              <1>       and    bx, bx ; 65536 -> 0
5905 000092A4 0F856BFFFFFF        <1>       jnz    loc_show_next_byte
5906 000092AA E954FFFFFF          <1>       jmp    loc_show_next_cluster
5907                              <1>
5908                              <1> loc_show_check_fat_cluster_size:
5909                              <1>       ; 17/02/2016
5910 000092AF 663B1D[C68A0100]    <1>       cmp    bx, [Show_ClusterSize] ; cluster size in bytes
5911 000092B6 0F8259FFFFFF        <1>       jb     loc_show_next_byte
5912 000092BC 66C705[C48A0100]00- <1>       mov    word [Show_ClusterPointer], 0
5912 000092C4 00                  <1>
5913                              <1>
5914 000092C5 A1[B88A0100]        <1>       mov    eax, [Show_Cluster]
5915                              <1>       ;mov   esi, [Show_LDDDT]
5916                              <1> loc_show_get_next_cluster:
5917 000092CA E86B350000          <1>       call   get_next_cluster
5918                              <1>       ;jc    loc_file_rw_cmd_failed
5919 000092CF 0F829BF7FFFF        <1>       jc     loc_run_cmd_failed
5920                              <1> loc_show_update_ccluster:
5921 000092D5 A3[B88A0100]        <1>       mov    [Show_Cluster], eax
5922 000092DA E924FFFFFF          <1>       jmp    loc_show_next_cluster
5923                              <1>
5924                              <1> loc_show_check_tab_space:
5925 000092DF 3C09                <1>       cmp    al, 09h
5926 000092E1 0F856EFFFFFF        <1>       jne   pass_show_dec_rowcount
5927                              <1> loc_show_put_tab_space:
5928 000092E7 8A3D[D6800100]      <1>       mov    bh, [ACTIVE_PAGE] ; [ptty]
5929 000092ED E8038CFFFF          <1>       call   get_cpos
5930                              <1>       ; dl = cursor column
5931 000092F2 80E207              <1>       and    dl, 7 ; 18/02/2016
5932                              <1>       ;shr   bh, 1 ; [ACTIVE_PAGE]
5933 000092F5 8A3D[D6800100]      <1>       mov    bh, [ACTIVE_PAGE]
5934 000092FB B307                <1>       mov    bl, 7 ; color attribute
5935                              <1> loc_show_put_space_chars:
5936 000092FD B020                <1>       mov    al, 20h ; space
5937                              <1>       ;mov   bh, [ACTIVE_PAGE] ; [ptty]
5938                              <1>       ;mov   bl, 7 ; color attribute
5939                              <1>       ;push  dx
5940 000092FF 52                  <1>       push   edx ; 29/12/2017
5941 00009300 E85E8FFFFF          <1>       call   _write_tty
5942 00009305 5A                  <1>       pop    edx ; 29/12/2017
5943                              <1>       ;pop   dx
5944                              <1>       ; 18/02/2016
5945 00009306 80FA07              <1>       cmp    dl, 7
5946 00009309 0F8353FFFFFF        <1>       jnb    loc_show_check_eof
5947 0000930F FEC2                <1>       inc    dl
5948 00009311 EBEA                <1>       jmp    short loc_show_put_space_chars
5949                              <1>
5950                              <1> check_filename:
5951                              <1>       ; 10/10/2016
5952                              <1>       ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
5953                              <1>       ; 07/08/2010 (FILE.ASM, 'proc_check_filename')
5954                              <1>       ; 10/07/2010
5955                              <1>       ; Derived from 'proc_check_filename'
5956                              <1>       ; in the old TRDOS.ASM (09/02/2005).
5957                              <1>       ;
5958                              <1>       ; INPUT ->
5959                              <1>       ;      ESI = Dot File Name Location
5960                              <1>       ; OUTPUT ->
5961                              <1>       ;      cf = 1 -> error code in AL
5962                              <1>       ;           AL = ERR_INV_FILE_NAME (=26)
5963                              <1>       ;               Invalid file name chars
5964                              <1>       ;      cf = 0 -> valid file name
5965                              <1>       ;
5966                              <1>       ;(EAX, ECX, EDI will be changed)
5967                              <1>
5968                              <1> check_invalid_filename_chars:
5969                              <1>       ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
5970                              <1>       ; 10/07/2010 (FILE.ASM, 'proc_check_invalid_filename_chars')
5971                              <1>       ; 10/02/2010
5972                              <1>       ; Derived from 'proc_check_invalid_filename_chars'
5973                              <1>       ; in the old TRDOS.ASM (09/02/2005).
5974                              <1>       ;
5975                              <1>       ; INPUT ->
5976                              <1>       ;      ESI = ASCIIZ FileName
5977                              <1>       ; OUTPUT ->
5978                              <1>       ;      cf = 1 -> invalid
5979                              <1>       ;      cf = 0 -> valid
5980                              <1>       ;
5981                              <1>       ;(EAX, ECX, EDI will be changed)
5982                              <1>
5983 00009313 56                  <1>       push   esi
5984                              <1>
5985 00009314 BF[543A0100]        <1>       mov   edi, invalid_fname_chars
5986 00009319 AC                  <1>       lodsb
5987                              <1> check_filename_next_char:
5988 0000931A B914000000          <1>       mov    ecx, sizeInvFnChars
5989 0000931F BF[543A0100]        <1>       mov    edi, invalid_fname_chars
5990                              <1> loc_scan_invalid_filename_char:
5991 00009324 AE                  <1>       scasb
5992 00009325 741F                <1>       je     short loc_invalid_filename_stc
5993 00009327 E2FB                <1>       loop   loc_scan_invalid_filename_char
```

```
5994 00009329 AC                <1>       lodsb
5995 0000932A 3C1F              <1>       cmp    al, 1Fh  ; 20h and above
5996 0000932C 77EC              <1>       ja     short check_filename_next_char
5997                            <1>
5998                            <1> check_filename_dot:
5999 0000932E 8B3424            <1>       mov    esi, [esp]
6000                            <1>
6001 00009331 B421              <1>       mov    ah, 21h
6002 00009333 B908000000        <1>       mov    ecx, 8
6003                            <1> loc_check_filename_next_char:
6004 00009338 AC                <1>       lodsb
6005 00009339 3C2E              <1>       cmp    al, 2Eh
6006 0000933B 7511              <1>       jne    short pass_check_fn_dot_check
6007                            <1> loc_check_filename_ext_0:
6008 0000933D AC                <1>       lodsb
6009 0000933E 38E0              <1>       cmp    al, ah ; 21h
6010 00009340 7205              <1>       jb     short loc_invalid_filename
6011 00009342 3C2E              <1>       cmp    al, 2Eh
6012 00009344 7519              <1>       jne    short loc_check_filename_ext_1
6013                            <1>
6014                            <1> loc_invalid_filename_stc:
6015                            <1> loc_check_fn_stc_rtn:
6016 00009346 F9                <1>       stc
6017                            <1> loc_invalid_filename:
6018                            <1>       ; 10/10/2016 (0Bh -> 26)
6019 00009347 B81A000000        <1>       mov    eax, ERR_INV_FILE_NAME ; (=26)
6020                            <1>       ; Invalid file name chars
6021                            <1> loc_check_fn_rtn:
6022 0000934C 5E                <1>       pop    esi
6023 0000934D C3                <1>       retn
6024                            <1>
6025                            <1> pass_check_fn_dot_check:
6026 0000934E 38E0              <1>       cmp    al, ah ; 21h
6027 00009350 7224              <1>       jb     short loc_check_fn_clc_rtn
6028 00009352 E2E4              <1>       loop   loc_check_filename_next_char
6029 00009354 AC                <1>       lodsb
6030 00009355 38E0              <1>       cmp    al, ah ; 21h
6031 00009357 721D              <1>       jb     short loc_check_fn_clc_rtn
6032 00009359 3C2E              <1>       cmp    al, 2Eh
6033 0000935B 75E9              <1>       jne    short loc_check_fn_stc_rtn
6034 0000935D EBDE              <1>       jmp    short loc_check_filename_ext_0
6035                            <1>
6036                            <1> loc_check_filename_ext_1:
6037 0000935F AC                <1>       lodsb
6038 00009360 38E0              <1>       cmp    al, ah ; 21h
6039 00009362 7212              <1>       jb     short loc_check_fn_clc_rtn
6040 00009364 3C2E              <1>       cmp    al, 2Eh
6041 00009366 74DE              <1>       je     short loc_check_fn_stc_rtn
6042 00009368 AC                <1>       lodsb
6043 00009369 38E0              <1>       cmp    al, ah ; 21h
6044 0000936B 7209              <1>       jb     short loc_check_fn_clc_rtn
6045 0000936D 3C2E              <1>       cmp    al, 2Eh
6046 0000936F 74D5              <1>       je     short loc_check_fn_stc_rtn
6047 00009371 AC                <1>       lodsb
6048 00009372 38E0              <1>       cmp    al, ah ; 21h
6049 00009374 73D0              <1>       jnb    short loc_check_fn_stc_rtn
6050                            <1>
6051                            <1> loc_check_fn_clc_rtn:
6052 00009376 5E                <1>       pop    esi
6053 00009377 F8                <1>       clc
6054 00009378 C3                <1>       retn
6055                            <1>
6056                            <1> loc_print_deleted_message:
6057 00009379 BE[413E0100]      <1>       mov    esi, Msg_Deleted
6058 0000937E E8C2DCFFFF        <1>       call   print_msg
6059                            <1>
6060                            <1>       ;clc
6061                            <1>
6062                            <1> loc_file_rw_restore_retn:
6063                            <1>       ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
6064                            <1>       ; 28/02/2010 (CMD_INTR.ASM)
6065                            <1> loc_file_rw_cmd_failed:
6066 00009383 9C                <1>       pushf
6067 00009384 E84FF7FFFF        <1>       call   restore_cdir_after_cmd_fail
6068 00009389 9D                <1>       popf
6069 0000938A 720D              <1>       jc     short loc_file_rw_check_write_fault
6070 0000938C C3                <1>       retn
6071                            <1>
6072                            <1> loc_permission_denied:
6073                            <1>       ; 27/02/2016
6074 0000938D BE[4E3E0100]      <1>       mov    esi, Msg_Permission_Denied
6075 00009392 E8AEDCFFFF        <1>       call   print_msg
6076 00009397 EBEA              <1>       jmp    short loc_file_rw_restore_retn
6077                            <1>
6078                            <1> loc_file_rw_check_write_fault:
6079                            <1>       ;cmp   al, 1Dh ; Write Fault
6080 00009399 3C12              <1>       cmp    al, 18 ; 05/11/2016
6081 0000939B 0F85D4F6FFFF      <1>       jne   loc_run_cmd_failed_cmp_al
6082 000093A1 BE[353C0100]      <1>       mov    esi, Msg_Not_Ready_Write_Err
6083                            <1>       ;call  print_msg
6084                            <1>       ;retn
6085 000093A6 E99ADCFFFF        <1>       jmp    print_msg
6086                            <1>
6087                            <1> make_directory:
6088                            <1>       ; 21/02/2016 (TRDOS 386 = TRDOS v2.0)
6089                            <1>       ; 12/03/2011 (CMD_INTR.ASM, 'cmp_cmd_mkdir')
6090                            <1>       ; 14/08/2010
6091                            <1>       ; 10/07/2010
6092                            <1>       ; 29/11/2009
6093                            <1>       ;
6094                            <1> get_mkdir_fchar:
6095                            <1>       ; esi = directory name
6096 000093AB 803E20            <1>       cmp    byte [esi], 20h
6097 000093AE 7701              <1>       ja     short loc_mkdir_parse_path_name
6098                            <1>
```

```
6099                                <1> loc_mkdir_nodirname_retn:
6100 000093B0 C3                   <1>         retn
6101                                <1>
6102                                <1> loc_mkdir_parse_path_name:
6103 000093B1 BF[0E8A0100]         <1>         mov    edi, FindFile_Drv
6104 000093B6 E86B1D0000           <1>         call   parse_path_name
6105 000093BB 0F8284F6FFFF         <1>         jc     loc_cmd_failed
6106                                <1>
6107                                <1> loc_mkdir_check_dirname_exists:
6108 000093C1 BE[508A0100]         <1>         mov    esi, FindFile_Name
6109 000093C6 803E20               <1>         cmp    byte [esi], 20h
6110 000093C9 0F8676F6FFFF         <1>         jna    loc_cmd_failed
6111 000093CF 8935[CC8A0100]       <1>         mov    [DelFile_FNPointer], esi
6112 000093D5 E839FFFFFF           <1>         call   check_filename
6113 000093DA 7259                 <1>         jc     short loc_mkdir_invalid_dir_name_chars
6114                                <1>
6115                                <1> loc_mkdir_drv:
6116 000093DC 8A35[6E810100]       <1>         mov    dh, [Current_Drv]
6117 000093E2 8835[CA880100]       <1>         mov    [RUN_CDRV], dh
6118                                <1>
6119 000093E8 8A15[0E8A0100]       <1>         mov    dl, [FindFile_Drv]
6120 000093EE 38F2                 <1>         cmp    dl, dh
6121 000093F0 7407                 <1>         je     short loc_mkdir_change_directory
6122                                <1>
6123 000093F2 E8BBE7FFFF           <1>         call   change_current_drive
6124 000093F7 728A                 <1>         jc     loc_file_rw_cmd_failed
6125                                <1>
6126                                <1> loc_mkdir_change_directory:
6127 000093F9 803D[0F8A0100]20     <1>         cmp    byte [FindFile_Directory], 20h
6128 00009400 7614                 <1>         jna    short loc_mkdir_find_directory
6129                                <1>
6130 00009402 FE05[55390100]       <1>         inc    byte [Restore_CDIR]
6131 00009408 BE[0F8A0100]         <1>         mov    esi, FindFile_Directory
6132 0000940D 30E4                 <1>         xor    ah, ah ; CD_COMMAND sign -> 0
6133 0000940F E8FC160000           <1>         call   change_current_directory
6134 00009414 722E                 <1>         jc     short loc_mkdir_check_error_code
6135                                <1>
6136                                <1> ;loc_mkdir_change_prompt_dir_string:
6137                                <1>         ;call  change_prompt_dir_string
6138                                <1>
6139                                <1> loc_mkdir_find_directory:
6140                                <1>         ;mov   esi, FindFile_Name
6141 00009416 8B35[CC8A0100]       <1>         mov    esi, [DelFile_FNPointer]
6142                                <1>         ;xor   eax, eax
6143 0000941C 6631C0               <1>         xor    ax, ax ; any name (dir, file, volume)
6144 0000941F E831FBFFFF           <1>         call   find_first_file
6145 00009424 721E                 <1>         jc     short loc_mkdir_check_error_code
6146                                <1>
6147                                <1> loc_mkdir_directory_found:
6148 00009426 BE[993D0100]         <1>         mov    esi, Msg_Name_Exists
6149 0000942B E815DCFFFF           <1>         call   print_msg
6150                                <1>
6151 00009430 E94EFFFFFF           <1>         jmp    loc_file_rw_restore_retn
6152                                <1>
6153                                <1> loc_mkdir_invalid_dir_name_chars:
6154 00009435 BE[6C3D0100]         <1>         mov    esi, Msg_invalid_name_chars
6155 0000943A E806DCFFFF           <1>         call   print_msg
6156                                <1>
6157 0000943F E93FFFFFFF           <1>         jmp    loc_file_rw_restore_retn
6158                                <1>
6159                                <1> loc_mkdir_check_error_code:
6160 00009444 3C02                 <1>         cmp    al, 2
6161                                <1>         ;je    short loc_mkdir_directory_not_found
6162 00009446 7406                 <1>         je     short loc_mkdir_ask_for_yes_no
6163 00009448 F9                   <1>         stc
6164 00009449 E935FFFFFF           <1>         jmp    loc_file_rw_cmd_failed
6165                                <1>
6166                                <1> loc_mkdir_directory_not_found:
6167                                <1> loc_mkdir_ask_for_yes_no:
6168 0000944E BE[BA3D0100]         <1>         mov    esi, Msg_DoYouWantMkdir
6169 00009453 E8EDDBFFFF           <1>         call   print_msg
6170 00009458 8B35[CC8A0100]       <1>         mov    esi, [DelFile_FNPointer]
6171 0000945E E8E2DBFFFF           <1>         call   print_msg
6172 00009463 BE[D93D0100]         <1>         mov    esi, Msg_YesNo
6173 00009468 E8D8DBFFFF           <1>         call   print_msg
6174                                <1>
6175 0000946D C605[E33D0100]20     <1>         mov    byte [Y_N_nextline], 20h
6176                                <1>
6177                                <1> loc_mkdir_ask_again:
6178 00009474 30E4                 <1>         xor    ah, ah
6179 00009476 E8607AFFFF           <1>         call   int16h
6180 0000947B 3C1B                 <1>         cmp    al, 1Bh
6181                                <1>         ;je    short loc_do_not_make_directory
6182 0000947D 7439                 <1>         je     short loc_mkdir_y_n_escape
6183 0000947F 24DF                 <1>         and    al, 0DFh ; y -> Y, n -> N
6184 00009481 3C59                 <1>         cmp    al, 'Y' ; 'yes'
6185 00009483 7404                 <1>         je     short loc_mkdir_yes_make_directory
6186 00009485 3C4E                 <1>         cmp    al, 'N' ; 'no'
6187 00009487 75EB                 <1>         jne    short loc_mkdir_ask_again
6188                                <1>
6189                                <1> loc_do_not_make_directory:
6190                                <1> loc_mkdir_yes_make_directory:
6191 00009489 E82E000000           <1>         call   y_n_answer ; 29/12/2017
6192                                <1>         ;cmp   al, 'Y' ; 'yes'
6193                                <1>         ;cmc
6194                                <1>         ;jnc   loc_file_rw_restore_retn
6195 0000948E 3C4E                 <1>         cmp    al, 'N' ; 'no'
6196 00009490 0F84EDFEFFFF         <1>         je     loc_file_rw_restore_retn
6197                                <1>
6198                                <1> loc_mkdir_call_make_sub_directory:
6199 00009496 8B35[CC8A0100]       <1>         mov    esi, [DelFile_FNPointer]
6200 0000949C B110                 <1>         mov    cl, 10h ; Directory attributes
6201 0000949E E8821D0000           <1>         call   make_sub_directory
6202                                <1> loc_rename_file_ok: ; 06/03/2016
6203 000094A3 0F82DAFEFFFF         <1>         jc     loc_file_rw_cmd_failed
```

```
6204                                <1> move_source_file_to_destination_OK:
6205 000094A9 BE[E73D0100]          <1>         mov    esi, Msg_OK
6206 000094AE E892DBFFFF            <1>         call   print_msg
6207 000094B3 E9CBFEFFFF            <1>         jmp    loc_file_rw_restore_retn
6208                                <1>
6209                                <1> loc_mkdir_y_n_escape:
6210 000094B8 B04E                  <1>         mov    al, 'N' ; 'no'
6211 000094BA EBCD                  <1>         jmp    short loc_do_not_make_directory
6212                                <1>
6213                                <1> y_n_answer:
6214                                <1>         ; 29/12/2017
6215 000094BC A2[E33D0100]          <1>         mov    [Y_N_nextline], al
6216                                <1>         ;push  ax
6217 000094C1 50                    <1>         push   eax
6218 000094C2 BE[E33D0100]          <1>         mov    esi, Y_N_nextline
6219 000094C7 E879DBFFFF            <1>         call   print_msg
6220 000094CC 58                    <1>         pop    eax
6221                                <1>         ;pop   ax
6222 000094CD C3                    <1>         retn
6223                                <1>
6224                                <1> delete_directory:
6225                                <1>         ; 29/12/2017
6226                                <1>         ; 15/10/2016
6227                                <1>         ; 01/03/2016, 06/03/2016
6228                                <1>         ; 27/02/2016, 28/02/2016, 29/02/2016
6229                                <1>         ; 26/02/2016 (TRDOS 386 = TRDOS v2.0)
6230                                <1>         ; 16/10/2010 (CMD_INTR.ASM, 'cmp_cmd_rmdir')
6231                                <1>         ; 05/06/2010
6232                                <1>         ;
6233                                <1> get_fchar:
6234                                <1>         ; esi = directory name
6235 000094CE 803E20                <1>         cmp    byte [esi], 20h
6236 000094D1 7701                  <1>           ja   short loc_rmdir_parse_path_name
6237                                <1>
6238                                <1> loc_rmdir_nodirname_retn:
6239 000094D3 C3                    <1>         retn
6240                                <1>
6241                                <1> loc_rmdir_parse_path_name:
6242 000094D4 BF[0E8A0100]          <1>         mov    edi, FindFile_Drv
6243 000094D9 E8481C0000            <1>         call   parse_path_name
6244 000094DE 0F8261F5FFFF          <1>         jc     loc_cmd_failed
6245                                <1>
6246                                <1> loc_rmdir_check_dirname_exists:
6247 000094E4 BE[508A0100]          <1>         mov    esi, FindFile_Name
6248 000094E9 803E20                <1>         cmp    byte [esi], 20h
6249 000094EC 0F8653F5FFFF          <1>         jna    loc_cmd_failed
6250 000094F2 8935[CC8A0100]        <1>         mov    [DelFile_FNPointer], esi
6251                                <1>
6252                                <1> loc_rmdir_drv:
6253 000094F8 8A35[6E810100]        <1>         mov    dh, [Current_Drv]
6254 000094FE 8835[CA880100]        <1>         mov    [RUN_CDRV], dh
6255                                <1>
6256 00009504 8A15[0E8A0100]        <1>         mov    dl, [FindFile_Drv]
6257 0000950A 38F2                  <1>         cmp    dl, dh
6258 0000950C 740B                  <1>         je     short loc_rmdir_change_directory
6259                                <1>
6260 0000950E E89FE6FFFF            <1>         call   change_current_drive
6261 00009513 0F826AFEFFFF          <1>         jc     loc_file_rw_cmd_failed
6262                                <1>
6263                                <1> loc_rmdir_change_directory:
6264 00009519 803D[0F8A0100]20      <1>         cmp    byte [FindFile_Directory], 20h
6265 00009520 7614                  <1>         jna    short loc_rmdir_find_directory
6266                                <1>
6267 00009522 FE05[55390100]        <1>         inc    byte [Restore_CDIR]
6268 00009528 BE[0F8A0100]          <1>         mov    esi, FindFile_Directory
6269 0000952D 30E4                  <1>         xor    ah, ah ; CD_COMMAND sign -> 0
6270 0000952F E8DC150000            <1>         call   change_current_directory
6271 00009534 7211                  <1>         jc     short loc_rmdir_check_error_code
6272                                <1>
6273                                <1> ;loc_rmdir_change_prompt_dir_string:
6274                                <1>         ;call  change_prompt_dir_string
6275                                <1>
6276                                <1> loc_rmdir_find_directory:
6277                                <1>         ;mov   esi, FindFile_Name
6278 00009536 8B35[CC8A0100]        <1>         mov    esi, [DelFile_FNPointer]
6279 0000953C 66B81008              <1>         mov    ax, 0810h ; Only directories
6280 00009540 E810FAFFFF            <1>         call   find_first_file
6281 00009545 730A                  <1>         jnc    short loc_rmdir_ambgfn_check
6282                                <1>
6283                                <1> loc_rmdir_check_error_code:
6284 00009547 3C02                  <1>         cmp    al, 2
6285 00009549 740B                  <1>         je     short loc_rmdir_directory_not_found
6286 0000954B F9                    <1>         stc
6287 0000954C E932FEFFFF            <1>         jmp    loc_file_rw_cmd_failed
6288                                <1>
6289                                <1> loc_rmdir_ambgfn_check:
6290 00009551 6621D2                <1>         and    dx, dx ; Ambiguous filename chars used sign (DX>0)
6291 00009554 740F                  <1>         jz     short loc_rmdir_directory_found
6292                                <1>
6293                                <1> loc_rmdir_directory_not_found:
6294 00009556 BE[573C0100]          <1>         mov    esi, Msg_Dir_Not_Found
6295 0000955B E8E5DAFFFF            <1>         call   print_msg
6296                                <1>
6297 00009560 E91EFEFFFF            <1>         jmp    loc_file_rw_restore_retn
6298                                <1>
6299                                <1> loc_rmdir_directory_found:
6300 00009565 80E307                <1>         and    bl, 07h ; Attributes
6301 00009568 0F851FFEFFFF          <1>         jnz    loc_permission_denied
6302                                <1>
6303                                <1> loc_rmdir_save_lnel: ; 28/02/2016
6304                                <1>          ;mov  bh, [LongName_EntryLength]
6305 0000956E 883D[D68A0100]        <1>         mov    [DelFile_LNEL], bh ; Long name entry length (if > 0)
6306                                <1>         ; edi = Directory Entry Offset (DirBuff)
6307                                <1>         ; esi = Directory Entry (FFF Structure)
6308                                <1>         ;mov   [DelFile_DirEntryAddr], edi ; not required
```

```
6309                                    <1>       ;mov   ax, [edi+20] ; First Cluster High Word
6310                                    <1>       ;shl   eax, 16
6311                                    <1>       ;mov   ax, [edi+26] ; First Cluster Low Word
6312                                    <1>       ; ROOT Dir First Cluster = 0
6313                                    <1>       ;cmp   eax, 2
6314                                    <1>       ;jb    loc_update_direntry_1
6315                                    <1>
6316                                    <1> pass_rmdir_fc_check:
6317 00009574 57                       <1>       push   edi ; * (29/02/2016)
6318                                    <1>
6319 00009575 BE[ED3D0100]             <1>       mov    esi, Msg_DoYouWantRmDir
6320 0000957A E8C6DAFFFF               <1>       call   print_msg
6321 0000957F 8B35[CC8A0100]           <1>       mov    esi, [DelFile_FNPointer]
6322 00009585 E8BBDAFFFF               <1>       call   print_msg
6323 0000958A BE[D93D0100]             <1>       mov    esi, Msg_YesNo
6324 0000958F E8B1DAFFFF               <1>       call   print_msg
6325                                    <1>
6326                                    <1> loc_rmdir_ask_again:
6327 00009594 30E4                     <1>       xor    ah, ah
6328 00009596 E84079FFFF               <1>       call   int16h
6329 0000959B 3C1B                     <1>       cmp    al, 1Bh
6330                                    <1>       ;je    short loc_do_not_delete_directory
6331 0000959D 7433                     <1>         je     loc_rmdir_y_n_escape ; 06/03/2016
6332 0000959F 24DF                     <1>       and    al, 0DFh
6333 000095A1 A2[E33D0100]             <1>       mov    [Y_N_nextline], al
6334 000095A6 3C59                     <1>       cmp    al, 'Y'
6335 000095A8 7404                     <1>       je     short loc_rmdir_yes_delete_directory
6336 000095AA 3C4E                     <1>       cmp    al, 'N'
6337 000095AC 75E6                     <1>       jne    short loc_rmdir_ask_again
6338                                    <1>
6339                                    <1> loc_do_not_delete_directory:
6340                                    <1> loc_rmdir_yes_delete_directory:
6341 000095AE E809FFFFFF               <1>       call   y_n_answer ; 29/12/2017
6342 000095B3 5F                       <1>       pop    edi ; * (29/02/2016)
6343                                    <1>       ;cmp   al, 'Y' ; 'yes'
6344                                    <1>       ;cmc
6345                                    <1>       ;jnc   loc_file_rw_restore_retn
6346 000095B4 3C4E                     <1>       cmp    al, 'N' ; 'no'
6347 000095B6 0F84C7FDFFFF             <1>       je     loc_file_rw_restore_retn
6348                                    <1>
6349                                    <1>       ; 29/12/2017
6350 000095BC E869000000               <1>       call   delete_sub_directory
6351 000095C1 7213                     <1>       jc     short loc_rmdir_cmd_failed
6352                                    <1>
6353                                    <1> loc_rmdir_ok:
6354 000095C3 BE[E73D0100]             <1>       mov    esi, Msg_OK
6355 000095C8 E878DAFFFF               <1>       call   print_msg
6356 000095CD E9B1FDFFFF               <1>       jmp    loc_file_rw_restore_retn
6357                                    <1>
6358                                    <1> loc_rmdir_y_n_escape:
6359 000095D2 B04E                     <1>       mov    al, 'N' ; 'no'
6360 000095D4 EBD8                     <1>       jmp     loc_do_not_delete_directory
6361                                    <1>
6362                                    <1> loc_rmdir_cmd_failed:
6363                                    <1>       ; 29/12/2017
6364 000095D6 09C0                     <1>       or     eax, eax ; EAX = 0 -> Directory not empty!
6365 000095D8 7426                     <1>       jz     short loc_rmdir_directory_not_empty
6366                                    <1>
6367                                    <1>       ; EAX > 0 -> Error code in AL (or AX or EAX)
6368                                    <1>
6369 000095DA 833D[8A880100]01         <1>       cmp    dword [FAT_ClusterCounter], 1
6370 000095E1 0F829CFDFFFF             <1>       jb     loc_file_rw_cmd_failed
6371 000095E7 F9                       <1>       stc
6372                                    <1> loc_rmdir_cmd_return:
6373                                    <1>       ; 01/03/2016
6374 000095E8 9C                       <1>       pushf
6375                                    <1>       ; ESI = Logical DOS Drive Description Table address
6376 000095E9 66BB00FF                 <1>       mov    bx, 0FF00h ; BH = FFh -> use ESI for Drive parameters
6377                                    <1>                       ; BL = 0 -> Recalculate free cluster count
6378 000095ED 50                       <1>       push   eax
6379 000095EE E8C3380000               <1>       call   calculate_fat_freespace
6380 000095F3 58                       <1>       pop    eax
6381 000095F4 9D                       <1>       popf
6382 000095F5 0F8288FDFFFF             <1>       jc     loc_file_rw_cmd_failed
6383 000095FB E983FDFFFF               <1>       jmp    loc_file_rw_restore_retn
6384                                    <1>
6385                                    <1> loc_rmdir_directory_not_empty:
6386 00009600 BE[0E3E0100]             <1>       mov    esi, Msg_Dir_Not_Empty
6387 00009605 E83BDAFFFF               <1>       call   print_msg
6388                                    <1>       ; 01/03/2016
6389 0000960A A1[8A880100]             <1>       mov    eax, [FAT_ClusterCounter]
6390 0000960F 09C0                     <1>       or     eax, eax ; 0 ?
6391 00009611 0F846CFDFFFF             <1>       jz     loc_file_rw_restore_retn
6392                                    <1>       ; ESI = Logical DOS Drive Description Table address
6393 00009617 66BB01FF                 <1>       mov    bx, 0FF01h ; BH = FFh -> use ESI for Drive parameters
6394                                    <1>                       ; BL = 1 -> add free clusters
6395 0000961B E896380000               <1>       call   calculate_fat_freespace
6396 00009620 09C9                     <1>       or     ecx, ecx
6397 00009622 0F845BFDFFFF             <1>       jz    loc_file_rw_restore_retn ; ecx = 0 -> OK
6398                                    <1>       ; ecx > 0 -> Error (Recalculation is needed)
6399 00009628 EBBE                     <1>       jmp    short loc_rmdir_cmd_return
6400                                    <1>
6401                                    <1>
6402                                    <1> delete_sub_directory:
6403                                    <1>       ; 29/12/2017
6404                                    <1>       ; (moved here from 'delete_directory' for 'sysrmdir' )
6405                                    <1>
6406                                    <1>       ; EDI = Directory buffer entry offset/address
6407                                    <1>
6408                                    <1> loc_rmdir_delete_short_name_check_dir_empty:
6409 0000962A 668B4714                 <1>       mov    ax, [edi+20] ; First Cluster High Word
6410 0000962E C1E010                   <1>       shl    eax, 16
6411 00009631 668B471A                 <1>       mov    ax, [edi+26] ; First Cluster Low Word
6412                                    <1>
6413                                    <1>       ;mov   [DelFile_FCluster], eax
```

```
                                <1>
                                <1>       ;;mov  bx, [DirBuff_EntryCounter]
                                <1>       ;mov   bx, [FindFile_DirEntryNumber] ; 27/02/2016
                                <1>       ;mov   [DelFile_EntryCounter], bx
                                <1>
00009635 29DB                   <1>       sub    ebx, ebx
                                <1>       ; 29/12/2017
00009637 891D[8A880100]         <1>       mov    [FAT_ClusterCounter], ebx ; 0 ; Reset
                                <1>
0000963D 8A3D[0E8A0100]         <1>       mov    bh, [FindFile_Drv]
00009643 BE00010900             <1>       mov    esi, Logical_DOSDisks
00009648 01DE                   <1>       add    esi, ebx
                                <1>
0000964A 66817F0CA101           <1>       cmp    word [edi+DirEntry_NTRes], 01A1h
00009650 745A                   <1>       je     short loc_rmdir_check_fs_directory
                                <1>
                                <1>       ;cmp   byte [esi+LD_FATType], 1
                                <1>       ;jb    short loc_rmdir_get__last_cluster_0
                                <1>
                                <1>       ; 29/12/2017
00009652 83F802                 <1>       cmp    eax, 2
00009655 7306                   <1>       jnb    short loc_rmdir_get_last_cluster_1
                                <1>       ; eax < 2
                                <1> loc_rmdir_get_last_cluster_0:
                                <1>       ;mov   eax, ERR_INV_FORMAT ; invalid format!
00009657 B813000000             <1>       mov    eax, ERR_NOT_DIR ; not a valid directory!
                                <1>       ;stc
0000965C C3                     <1>       retn
                                <1>
                                <1> loc_rmdir_get_last_cluster_1:
0000965D 807E0303               <1>       cmp    byte [esi+LD_FATType], 3 ; FAT32
00009661 750C                   <1>       jne    short loc_rmdir_get_last_cluster_2
                                <1>
                                <1>       ; is it root directory ?
00009663 3B4632                 <1>       cmp    eax, [esi+LD_BPB+BPB_RootClus]
00009666 7507                   <1>       jne    short loc_rmdir_get_last_cluster_2
                                <1>
                                <1>       ; root directory can not be deleted !!
                                <1> loc_rmdir_permission_denied:
00009668 B80B000000             <1>       mov    eax, ERR_PERM_DENIED ; permission denied!
0000966D F9                     <1>       stc
0000966E C3                     <1>       retn
                                <1>
                                <1> loc_rmdir_get_last_cluster_2:
                                <1>       ; 29/12/2017
0000966F A3[D08A0100]           <1>       mov    [DelFile_FCluster], eax
                                <1>
                                <1>       ;mov   dx, [DirBuff_EntryCounter]
00009674 668B15[888A0100]       <1>       mov    dx, [FindFile_DirEntryNumber] ; 27/02/2016
0000967B 668915[D48A0100]       <1>       mov    [DelFile_EntryCounter], dx
                                <1>
00009682 8B15[99880100]         <1>       mov    edx, [DirBuff_Cluster]
00009688 8915[008B0100]         <1>       mov    [RmDir_ParentDirCluster], edx
                                <1>
0000968E 893D[FC8A0100]         <1>       mov    [RmDir_DirEntryOffset], edi
                                <1>
                                <1>       ; 01/03/2016
                                <1>       ;mov   dword [FAT_ClusterCounter], 0 ; Reset
                                <1>
                                <1> loc_rmdir_get_last_cluster_3:
00009694 E89C390000             <1>       call   get_last_cluster
                                <1>         ;jc  loc_rmdir_cmd_failed
00009699 721E                   <1>       jc     short loc_delete_sub_dir_retn ; 29/12/2017
                                <1>
0000969B 3B05[D08A0100]         <1>       cmp    eax, [DelFile_FCluster]
000096A1 7517                   <1>       jne    short loc_rmdir_multi_dir_clusters
                                <1>
000096A3 C605[FB8A0100]00       <1>       mov    byte [RmDir_MultiClusters], 0
000096AA EB15                   <1>       jmp    short pass_rmdir_multi_dir_clusters
                                <1>
                                <1> loc_rmdir_check_fs_directory:
                                <1>       ; 29/12/2017
000096AC 807E04A1               <1>       cmp    byte [esi+LD_FSType], 0A1h
000096B0 75B6                   <1>       jne    short loc_rmdir_permission_denied
                                <1>
                                <1> loc_rmdir_delete_fs_directory:
000096B2 E876130000             <1>       call   delete_fs_directory
                                <1>       ;jnc   loc_print_deleted_message
000096B7 7300                   <1>       jnc    short loc_delete_sub_dir_retn ; 29/12/2017
                                <1>
                                <1>       ; EAX=0 -> Directory not empty !
                                <1>       ; EAX>0 -> Disk r/w error or another (misc) error
                                <1>
                                <1>       ;or    eax, eax
                                <1>       ;jz    loc_rmdir_directory_not_empty_2
                                <1>       ;;stc
                                <1>       ;;jmp  loc_file_rw_cmd_failed
                                <1>
                                <1> loc_delete_sub_dir_retn:
000096B9 C3                     <1>       retn
                                <1>
                                <1> loc_rmdir_multi_dir_clusters:
000096BA C605[FB8A0100]01       <1>       mov    byte [RmDir_MultiClusters], 1
                                <1>
                                <1> pass_rmdir_multi_dir_clusters:
000096C1 A3[048B0100]           <1>       mov    [RmDir_DirLastCluster], eax
000096C6 890D[088B0100]         <1>       mov    [RmDir_PreviousCluster], ecx
                                <1>
                                <1> loc_rmdir_load_fat_sub_directory:
000096CC E84F330000             <1>       call   load_FAT_sub_directory
                                <1>       ;jc    loc_rmdir_cmd_failed
000096D1 72E6                   <1>       jc     short loc_delete_sub_dir_retn
                                <1>
                                <1> loc_rmdir_find_last_dir_entry:
000096D3 56                     <1>       push   esi
```

```
6519 000096D4 BE[F2890100]        <1>       mov    esi, Dir_File_Name
6520 000096D9 C6062A              <1>       mov    byte [esi], '*'
6521 000096DC C646082A            <1>       mov    byte [esi+8], '*'
6522 000096E0 31DB                <1>       xor    ebx, ebx ; Entry offset  = 0
6523                              <1> loc_rmdir_find_last_dir_entry_next:
6524 000096E2 66B80008            <1>       mov    ax, 0800h ; Except volume/long names
6525 000096E6 6631C9              <1>       xor    cx, cx ; 0 = Find a valid file or dir name
6526 000096E9 E879170000          <1>       call   find_directory_entry
6527 000096EE 7225                <1>       jc     short loc_rmdir_empty_dir_cluster
6528 000096F0 83FB01              <1>       cmp    ebx, 1
6529 000096F3 771B                <1>       ja     short loc_rmdir_directory_not_empty_1
6530                              <1> loc_rmdir_dot_entry_check:
6531 000096F5 80FD2E              <1>       cmp    ch, '.' ; The first char of the dir entry
6532 000096F8 7516                <1>       jne    short loc_rmdir_directory_not_empty_1
6533 000096FA 08DB                <1>       or     bl, bl
6534 000096FC 7506                <1>       jnz    short loc_rmdir_dotdot_entry_check
6535 000096FE 807F0120            <1>       cmp    byte [edi+1], 20h
6536 00009702 EB06                <1>       jmp    short pass_rmdir_dot_entry_check
6537                              <1>
6538                              <1> loc_rmdir_dotdot_entry_check:
6539 00009704 66817F012E20        <1>       cmp    word [edi+1], '. '
6540                              <1> pass_rmdir_dot_entry_check:
6541 0000970A 7504                <1>       jne    short loc_rmdir_directory_not_empty_1
6542 0000970C FEC3                <1>       inc    bl
6543 0000970E EBD2                <1>       jmp    short loc_rmdir_find_last_dir_entry_next
6544                              <1>
6545                              <1> loc_rmdir_directory_not_empty_1:
6546 00009710 58                  <1>       pop    eax ; pushed esi
6547 00009711 31C0                <1>       xor    eax, eax ; 0
6548                              <1> loc_rmdir_directory_not_empty_2:
6549                              <1> loc_delete_sub_dir_stc_retn:
6550 00009713 F9                  <1>       stc
6551 00009714 C3                  <1>       retn
6552                              <1>
6553                              <1> loc_rmdir_empty_dir_cluster:
6554 00009715 5E                  <1>       pop    esi
6555                              <1>
6556                              <1> loc_rmdir_set_prev_cluster_dir_last_cluster:
6557 00009716 803D[FB8A0100]00    <1>       cmp    byte [RmDir_MultiClusters], 0
6558 0000971D 7613                <1>       jna    short loc_rmdir_unlink_dir_last_cluster
6559                              <1>
6560 0000971F A1[088B0100]        <1>       mov    eax, [RmDir_PreviousCluster]
6561                              <1>       ;xor   ecx, ecx
6562 00009724 49                  <1>       dec    ecx ; FFFFFFFFh
6563 00009725 E83A340000          <1>       call   update_cluster
6564 0000972A 7306                <1>       jnc    short loc_rmdir_unlink_dir_last_cluster
6565                              <1>
6566                              <1>       ; 01/03/2016
6567                              <1>       ;cmp   eax, 1  ; eax = 0 -> end of cluster chain
6568                              <1>       ;cmc
6569                              <1>       ;jc    short loc_rmdir_cmd_failed
6570                              <1>       ;jmp   short loc_rmdir_save_fat_buffer
6571                              <1>       ; 29/12/2017
6572 0000972C 21C0                <1>       and    eax, eax
6573 0000972E 75E3                <1>       jnz    short loc_delete_sub_dir_stc_retn
6574 00009730 EB12                <1>       jmp    short loc_rmdir_save_fat_buffer
6575                              <1>
6576                              <1> loc_rmdir_unlink_dir_last_cluster:
6577 00009732 A1[048B0100]        <1>       mov    eax, [RmDir_DirLastCluster]
6578 00009737 31C9                <1>       xor    ecx, ecx ; 0
6579 00009739 E826340000          <1>       call   update_cluster
6580 0000973E 7327                <1>       jnc    short loc_rmdir_unlink_stc_retn_0Bh
6581                              <1>       ; Because of it is the last cluster
6582                              <1>       ; 'update_cluster' must return with eocc error
6583 00009740 09C0                <1>       or     eax, eax
6584                              <1>       ;jz    short loc_rmdir_save_fat_buffer ; eocc
6585                              <1>       ;stc
6586                              <1>         ;jmp     short loc_rmdir_cmd_failed
6587                              <1>       ; 29/12/2017
6588 00009742 75CF                <1>       jnz    short loc_delete_sub_dir_stc_retn
6589                              <1>
6590                              <1> loc_rmdir_save_fat_buffer:
6591 00009744 803D[82880100]02    <1>       cmp    byte [FAT_BuffValidData], 2
6592 0000974B 7528                <1>       jne    short loc_rmdir_calculate_FAT_freespace
6593 0000974D E8CF360000          <1>       call   save_fat_buffer
6594                              <1>       ;jc    short loc_rmdir_cmd_failed
6595                              <1>       ; 29/12/2017
6596 00009752 7219                <1>       jc     short loc_rmdir_unlink_error_retn
6597                              <1>
6598                              <1>       ; 01/03/2016
6599 00009754 803D[FB8A0100]00    <1>       cmp    byte [RmDir_MultiClusters], 0
6600 0000975B 7618                <1>       jna    short loc_rmdir_calculate_FAT_freespace
6601                              <1>
6602 0000975D A1[D08A0100]        <1>       mov    eax, [DelFile_FCluster]
6603 00009762 E92DFFFFFF          <1>       jmp   loc_rmdir_get_last_cluster_3
6604                              <1>
6605                              <1> loc_rmdir_unlink_stc_retn_0Bh:
6606                              <1>       ; 15/10/2016 (0Bh -> 28)
6607 00009767 B81C000000          <1>       mov    eax, ERR_INV_FORMAT ; 28 = Invalid format
6608                              <1> loc_rmdir_unlink_stc_retn:
6609 0000976C F9                  <1>       stc
6610                              <1> loc_rmdir_unlink_error_retn:
6611 0000976D C3                  <1>       retn
6612                              <1>
6613                              <1> loc_rmdir_delete_short_name_invalid_data:
6614 0000976E B81D000000          <1>       mov    eax, 29 ; Invalid data (15/10/2016)
6615                              <1>       ;stc
6616                              <1>       ;jmp   loc_rmdir_cmd_failed
6617                              <1>       ; 29/12/2017
6618 00009773 EBF7                <1>       jmp    short loc_rmdir_unlink_stc_retn
6619                              <1>
6620                              <1> loc_rmdir_calculate_FAT_freespace:
6621                              <1>       ;mov   eax, [FAT_ClusterCounter]
6622                              <1>       ; 29/12/2017
6623 00009775 29C0                <1>       sub    eax, eax ; 0
```

```
6624 00009777 8705[8A880100]      <1>       xchg   eax, [FAT_ClusterCounter]
6625                              <1>       ;
6626 0000977D 66BB01FF            <1>       mov    bx, 0FF01h
6627                              <1>       ; BL = 1 -> Add EAX to free space count
6628                              <1>       ; BH = FFh ->
6629                              <1>       ; ESI = Logical DOS Drive Description Table address
6630 00009781 E830370000          <1>       call   calculate_fat_freespace
6631                              <1>
6632 00009786 21C9                <1>       and    ecx, ecx ; ecx = 0 -> valid free sector count
6633 00009788 7409                <1>       jz     short loc_rmdir_delete_short_name_continue
6634                              <1>
6635                              <1> loc_rmdir_recalculate_FAT_freespace:
6636 0000978A 66BB00FF            <1>       mov   bx, 0FF00h ; BL = 0 -> Recalculate free space
6637 0000978E E823370000          <1>       call   calculate_fat_freespace
6638                              <1>
6639                              <1> loc_rmdir_delete_short_name_continue:
6640 00009793 A1[008B0100]        <1>       mov    eax, [RmDir_ParentDirCluster]
6641 00009798 83F802              <1>       cmp    eax, 2
6642 0000979B 7309                <1>       jnb    short loc_rmdir_del_short_name_load_sub_dir
6643 0000979D E8F3310000          <1>       call   load_FAT_root_directory
6644                              <1>       ;jc    loc_file_rw_cmd_failed
6645                              <1>       ; 29/12/2017
6646 000097A2 72C9                <1>       jc     short loc_rmdir_unlink_error_retn
6647 000097A4 EB07                <1>       jmp    short loc_rmdir_del_short_name_ld_chk_fclust
6648                              <1>
6649                              <1> loc_rmdir_del_short_name_load_sub_dir:
6650 000097A6 E875320000          <1>       call   load_FAT_sub_directory
6651                              <1>       ;jc    loc_file_rw_cmd_failed
6652                              <1>       ; 29/12/2017
6653 000097AB 72C0                <1>       jc     short loc_rmdir_unlink_error_retn
6654                              <1>
6655                              <1> loc_rmdir_del_short_name_ld_chk_fclust:
6656 000097AD 0FB73D[FC8A0100]    <1>       movzx  edi, word [RmDir_DirEntryOffset]
6657 000097B4 81C700000800        <1>       add    edi, Directory_Buffer
6658                              <1>
6659 000097BA 668B4714            <1>       mov    ax, [edi+20] ; First Cluster High Word
6660 000097BE C1E010              <1>       shl    eax, 16
6661 000097C1 668B471A            <1>       mov    ax, [edi+26] ; First Cluster Low Word
6662                              <1>         ; Not necessary...
6663 000097C5 3B05[D08A0100]      <1>       cmp    eax, [DelFile_FCluster]
6664 000097CB 75A1                <1>       jne    short loc_rmdir_delete_short_name_invalid_data
6665                              <1>       ;
6666 000097CD C607E5              <1>       mov    byte [edi], 0E5h ; 'Deleted' sign
6667                              <1>       ; 27/02/2016
6668                              <1>       ; TRDOS v1 has a bug here! it does not set
6669                              <1>       ; 'DirBuff_ValidData' to 2; as result of this bug,
6670                              <1>       ; 'save_directory_buffer' would not save the change !
6671 000097D0 C605[94880100]02    <1>       mov    byte [DirBuff_ValidData], 2 ; change sign
6672                              <1>       ;
6673 000097D7 E8AE1D0000          <1>       call   save_directory_buffer
6674                              <1>       ;jc    loc_file_rw_cmd_failed
6675                              <1>       ; 29/12/2017
6676 000097DC 728F                <1>       jc     short loc_rmdir_unlink_error_retn
6677                              <1>
6678                              <1> loc_rmdir_del_long_name:
6679 000097DE 0FB615[D68A0100]    <1>       movzx  edx, byte [DelFile_LNEL]
6680 000097E5 08D2                <1>       or     dl, dl
6681 000097E7 7410                <1>       jz     short loc_rmdir_update_parent_dir_lmdt
6682                              <1>
6683 000097E9 0FB705[D48A0100]    <1>       movzx  eax, word [DelFile_EntryCounter]
6684 000097F0 29D0                <1>       sub    eax, edx
6685                              <1>       ; 29/12/2017
6686 000097F2 7205                <1>       jc     short loc_rmdir_update_parent_dir_lmdt
6687                              <1>
6688                              <1>       ; EAX = Directory Entry Number of the long name last entry
6689 000097F4 E8EF1E0000          <1>       call   delete_longname
6690                              <1>
6691                              <1> loc_rmdir_update_parent_dir_lmdt:
6692 000097F9 E8271E0000          <1>       call   update_parent_dir_lmdt
6693                              <1>       ;jc    short loc_file_rw_cmd_failed
6694                              <1>       ; 29/12/2017
6695                              <1>       ;jc    short loc_rmdir_unlink_error_retn
6696                              <1>
6697                              <1> loc_delete_sub_directory_ok:
6698                              <1>       ; 29/12/2017
6699 000097FE 31C0                <1>       xor    eax, eax ;  0 ;  cf = 0
6700 00009800 C3                  <1>       retn
6701                              <1>
6702                              <1>
6703                              <1> delete_file:
6704                              <1>       ; 29/02/2016
6705                              <1>       ; 28/02/2016 (TRDOS 386 = TRDOS v2.0)
6706                              <1>       ; 09/08/2010 (CMD_INTR.ASM, 'cmp_cmd_del')
6707                              <1>       ; 28/02/2010
6708                              <1>
6709                              <1> get_delfile_fchar:
6710                              <1>       ; esi = file name
6711 00009801 803E20              <1>       cmp    byte [esi], 20h
6712 00009804 7701                <1>       ja     short loc_delfile_parse_path_name
6713                              <1>
6714                              <1> loc_delfile_nofilename_retn:
6715 00009806 C3                  <1>       retn
6716                              <1>
6717                              <1> loc_delfile_parse_path_name:
6718 00009807 BF[0E8A0100]        <1>       mov    edi, FindFile_Drv
6719 0000980C E815190000          <1>       call   parse_path_name
6720 00009811 0F822EF2FFFF        <1>       jc     loc_cmd_failed
6721                              <1>
6722                              <1> loc_delfile_check_filename_exists:
6723 00009817 BE[508A0100]        <1>       mov    esi, FindFile_Name
6724 0000981C 803E20              <1>       cmp    byte [esi], 20h
6725 0000981F 0F8620F2FFFF        <1>       jna    loc_cmd_failed
6726 00009825 8935[CC8A0100]      <1>       mov    [DelFile_FNPointer], esi
6727                              <1>
6728                              <1> loc_delfile_drv:
```

```
6729 0000982B 8A15[0E8A0100]      <1>       mov   dl, [FindFile_Drv]
6730 00009831 8A35[6E810100]      <1>       mov   dh, [Current_Drv]
6731 00009837 8835[CA880100]      <1>       mov   [RUN_CDRV], dh
6732 0000983D 38F2                <1>       cmp   dl, dh
6733 0000983F 740B                <1>       je    short loc_delfile_change_directory
6734                              <1>
6735 00009841 E86CE3FFFF          <1>       call  change_current_drive
6736 00009846 0F8237FBFFFF        <1>       jc    loc_file_rw_cmd_failed
6737                              <1>
6738                              <1> loc_delfile_change_directory:
6739 0000984C 803D[0F8A0100]20    <1>       cmp   byte [FindFile_Directory], 20h
6740 00009853 7618                <1>       jna   short loc_delfile_find
6741                              <1>
6742 00009855 FE05[55390100]      <1>       inc   byte [Restore_CDIR]
6743 0000985B BE[0F8A0100]        <1>       mov   esi, FindFile_Directory
6744 00009860 30E4                <1>       xor   ah, ah ; CD_COMMAND sign -> 0
6745 00009862 E8A9120000          <1>       call  change_current_directory
6746 00009867 0F8216FBFFFF        <1>       jc    loc_file_rw_cmd_failed
6747                              <1>
6748                              <1> ;loc_delfile_change_prompt_dir_string:
6749                              <1>       ;call  change_prompt_dir_string
6750                              <1>
6751                              <1> loc_delfile_find:
6752                              <1>       ;mov   esi, FindFile_Name
6753 0000986D 8B35[CC8A0100]      <1>       mov   esi, [DelFile_FNPointer]
6754 00009873 66B80018            <1>       mov   ax, 1800h ; Except volume label and dirs
6755 00009877 E8D9F6FFFF          <1>       call  find_first_file
6756 0000987C 0F8201FBFFFF        <1>       jc    loc_file_rw_cmd_failed
6757                              <1>
6758                              <1> loc_delfile_ambgfn_check:
6759 00009882 6621D2              <1>       and   dx, dx ; Ambiguous filename chars used sign (DX>0)
6760 00009885 740B                <1>       jz    short loc_delfile_found
6761                              <1>
6762                              <1> loc_file_not_found:
6763 00009887 B802000000          <1>       mov   eax, 2 ; File not found sign
6764 0000988C F9                  <1>       stc
6765 0000988D E9F1FAFFFF          <1>       jmp   loc_file_rw_cmd_failed
6766                              <1>
6767                              <1> loc_delfile_found:
6768 00009892 80E307              <1>       and   bl, 07h ; Attributes
6769 00009895 0F85F2FAFFFF        <1>         jnz     loc_permission_denied
6770                              <1>
6771                              <1> ;loc_delfile_found_save_lnel:
6772                              <1> ;     mov   [DelFile_LNEL], bh ; Long name entry length (if > 0)
6773                              <1>
6774                              <1> loc_delfile_ask_for_delete:
6775 0000989B 57                  <1>       push  edi ; * (29/02/2016)
6776                              <1>
6777 0000989C BE[253E0100]        <1>       mov   esi, Msg_DoYouWantDelete
6778 000098A1 E89FD7FFFF          <1>       call  print_msg
6779 000098A6 8B35[CC8A0100]      <1>       mov   esi, [DelFile_FNPointer]
6780 000098AC E894D7FFFF          <1>       call  print_msg
6781 000098B1 BE[D93D0100]        <1>       mov   esi, Msg_YesNo
6782 000098B6 E88AD7FFFF          <1>       call  print_msg
6783                              <1>
6784                              <1> loc_delfile_ask_again:
6785 000098BB 30E4                <1>       xor   ah, ah
6786 000098BD E81976FFFF          <1>       call  int16h
6787 000098C2 3C1B                <1>       cmp   al, 1Bh
6788                              <1>       ;je   short loc_do_not_delete_file
6789 000098C4 7449                <1>       je    short loc_delfile_y_n_escape ; 06/03/2016
6790 000098C6 24DF                <1>       and   al, 0DFh
6791 000098C8 A2[E33D0100]        <1>       mov   [Y_N_nextline], al
6792 000098CD 3C59                <1>       cmp   al, 'Y'
6793 000098CF 7404                <1>       je    short loc_yes_delete_file
6794 000098D1 3C4E                <1>       cmp   al, 'N'
6795 000098D3 75E6                <1>       jne   short loc_delfile_ask_again
6796                              <1>
6797                              <1> loc_do_not_delete_file:
6798                              <1> loc_yes_delete_file:
6799 000098D5 E8E2FBFFFF          <1>       call  y_n_answer ; 29/12/2017
6800 000098DA 5F                  <1>       pop   edi ; * (29/02/2016)
6801                              <1>       ;cmp  al, 'Y' ; 'yes'
6802                              <1>       ;cmc
6803                              <1>         ;jnc loc_file_rw_restore_retn
6804 000098DB 3C4E                <1>       cmp   al, 'N' ; 'no'
6805 000098DD 0F84A0FAFFFF        <1>         je  loc_file_rw_restore_retn
6806                              <1>
6807                              <1> loc_delete_file:
6808 000098E3 8A3D[0E8A0100]      <1>       mov   bh, [FindFile_Drv]
6809                              <1>       ;mov  bl, [DelFile_LNEL]
6810 000098E9 8A1D[5D8A0100]      <1>       mov   bl, [FindFile_LongNameEntryLength]
6811                              <1>       ;mov  cx, [DirBuff_EntryCounter]
6812 000098EF 668B0D[888A0100]    <1>       mov   cx, [FindFile_DirEntryNumber]
6813                              <1>       ; (*) EDI = Directory buffer entry offset/address
6814 000098F6 E8D71F0000          <1>       call  remove_file ; (FILE.ASM, 'proc_delete_file')
6815 000098FB 0F8378FAFFFF        <1>       jnc   loc_print_deleted_message
6816                              <1>
6817                              <1>       ;cmp  al, 05h
6818 00009901 3C0B                <1>       cmp   al, ERR_PERM_DENIED  ; 29/12/2017 (5 -> 11)
6819 00009903 0F8484FAFFFF        <1>       je    loc_permission_denied
6820 00009909 F9                  <1>       stc
6821 0000990A E974FAFFFF          <1>       jmp   loc_file_rw_cmd_failed
6822                              <1>
6823                              <1> loc_delfile_y_n_escape:
6824 0000990F B04E                <1>       mov   al, 'N' ; 'no'
6825 00009911 EBC2                <1>       jmp   short loc_do_not_delete_file
6826                              <1>
6827                              <1> set_file_attributes:
6828                              <1>       ; 06/03/2016
6829                              <1>       ; 04/03/2016 (TRDOS 386 = TRDOS v2.0)
6830                              <1>       ; 10/07/2010 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_attrib')
6831                              <1>       ; 23/05/2010
6832                              <1>       ; 17/12/2000 (P2000.ASM)
6833                              <1>
```

```
                                <1>       ; esi = file or directory name
00009913 6631C0                 <1>       xor    ax, ax
00009916 66A3[763E0100]         <1>       mov    [Attr_Chars], ax
0000991C A2[248B0100]           <1>       mov    [Attributes], al
                                <1>
                                <1> get_attrib_fchar:
                                <1>       ; esi = file name
00009921 8A06                   <1>       mov    al, [esi]
00009923 3C20                   <1>       cmp    al, 20h
00009925 7623                   <1>       jna    short loc_attr_file_nofilename_retn
                                <1>
                                <1> loc_scan_attrib_params:
00009927 3C2D                   <1>       cmp    al, '-'
00009929 0F871C010000           <1>       ja     loc_attr_file_parse_path_name
0000992F 7408                   <1>       je     short loc_attr_space
                                <1>
00009931 3C2B                   <1>       cmp    al, '+'
00009933 0F850CF1FFFF           <1>       jne    loc_cmd_failed
                                <1>
                                <1> loc_attr_space:
00009939 8A6601                 <1>       mov    ah, [esi+1]
0000993C 80FC20                 <1>       cmp    ah, 20h
0000993F 770A                   <1>       ja     short pass_attr_space
00009941 0F82FEF0FFFF           <1>       jb     loc_cmd_failed
00009947 46                     <1>       inc    esi
00009948 EBEF                   <1>       jmp    short loc_attr_space
                                <1>
                                <1> loc_attr_file_nofilename_retn:
0000994A C3                     <1>       retn
                                <1>
                                <1> pass_attr_space:
0000994B 80E4DF                 <1>       and    ah, 0DFh
0000994E 80FC53                 <1>       cmp    ah, 'S'
00009951 0F87EEF0FFFF           <1>       ja     loc_cmd_failed
00009957 7204                   <1>       jb     short pass_attr_system
00009959 B404                   <1>       mov    ah, 04h   ; System
0000995B EB21                   <1>       jmp    short pass_attr_archive
                                <1>
                                <1> pass_attr_system:
0000995D 80FC48                 <1>       cmp    ah, 'H'
00009960 7706                   <1>       ja     short pass_attr_hidden
00009962 7213                   <1>       jb     short pass_attr_read_only
00009964 B402                   <1>       mov    ah, 02h    ; Hidden
00009966 EB16                   <1>       jmp    short pass_attr_archive
                                <1>
                                <1> pass_attr_hidden:
00009968 80FC52                 <1>       cmp    ah, 'R'
0000996B 0F87D4F0FFFF           <1>       ja     loc_cmd_failed
00009971 7204                   <1>       jb     short pass_attr_read_only ; Read only
00009973 B401                   <1>       mov    ah, 01h
00009975 EB07                   <1>       jmp    short pass_attr_archive
                                <1>
                                <1> pass_attr_read_only:
00009977 80FC41                 <1>       cmp    ah, 'A'
0000997A 753B                   <1>       jne    short loc_chk_attr_enter
0000997C B420                   <1>       mov    ah, 20h    ; Archive
                                <1>
                                <1> pass_attr_archive:
0000997E 3C2D                   <1>       cmp    al, '-'
00009980 7508                   <1>       jne    short pass_reducing_attributes
00009982 0825[763E0100]         <1>       or     [Attr_Chars], ah
00009988 EB06                   <1>       jmp    short loc_change_attributes_inc
                                <1>
                                <1> pass_reducing_attributes:
0000998A 0825[773E0100]         <1>       or     [Attr_Chars+1], ah
                                <1>
                                <1> loc_change_attributes_inc:
00009990 46                     <1>       inc    esi
00009991 8A6601                 <1>       mov    ah, [esi+1]
00009994 80FC20                 <1>       cmp    ah, 20h
00009997 7227                   <1>       jb     short pass_change_attr
00009999 74F5                   <1>       je     short loc_change_attributes_inc
0000999B 80FC2D                 <1>       cmp    ah, '-'
0000999E 770D                   <1>       ja     short loc_chk_next_attr_char1
000099A0 7405                   <1>       je     short loc_chk_next_attr_char0
000099A2 80FC2B                 <1>       cmp    ah, '+'
000099A5 7506                   <1>       jne    short loc_chk_next_attr_char1
                                <1>
                                <1> loc_chk_next_attr_char0:
000099A7 46                     <1>       inc    esi
000099A8 668B06                 <1>       mov    ax, [esi]
000099AB EB9E                   <1>       jmp    short pass_attr_space
                                <1>
                                <1> loc_chk_next_attr_char1:
000099AD 803E2D                 <1>       cmp    byte [esi], '-'
000099B0 7799                   <1>       ja     short pass_attr_space
000099B2 E988000000             <1>         jmp     loc_attr_file_check_fname_fchar
                                <1>
                                <1> loc_chk_attr_enter:
000099B7 80FC0D                 <1>       cmp    ah, 0Dh
000099BA 0F8585F0FFFF           <1>       jne    loc_cmd_failed
                                <1>
                                <1> pass_change_attr:
000099C0 A0[763E0100]           <1>       mov    al, [Attr_Chars]
000099C5 F6D0                   <1>       not    al
000099C7 2005[248B0100]         <1>       and    [Attributes], al
000099CD A0[773E0100]           <1>       mov    al, [Attr_Chars+1]
000099D2 0805[248B0100]         <1>       or     [Attributes], al
                                <1>
                                <1> loc_show_attributes:
000099D8 BE[D7430100]           <1>       mov    esi, nextline
000099DD E863D6FFFF             <1>       call   print_msg
                                <1>
                                <1> loc_show_attributes_no_nextline:
000099E2 C705[763E0100]4E4F-    <1>       mov    dword [Attr_Chars], 'NORM'
```

```
6938 000099EA 524D                <1>
6939 000099EC 66C705[7A3E0100]41- <1>       mov    word [Attr_Chars+4], 'AL'
6939 000099F4 4C                  <1>
6940 000099F5 BE[763E0100]        <1>       mov    esi, Attr_Chars
6941 000099FA A0[248B0100]        <1>       mov    al, [Attributes]
6942 000099FF A804                <1>       test   al, 04h
6943 00009A01 7406                <1>       jz     short pass_put_attr_s
6944 00009A03 66C7065300          <1>       mov    word [esi], 0053h     ; S
6945 00009A08 46                  <1>       inc    esi
6946                              <1>
6947                              <1> pass_put_attr_s:
6948 00009A09 A802                <1>       test   al, 02h
6949 00009A0B 7406                <1>       jz     short pass_put_attr_h
6950 00009A0D 66C7064800          <1>       mov    word [esi], 0048h     ; H
6951 00009A12 46                  <1>       inc    esi
6952                              <1>
6953                              <1> pass_put_attr_h:
6954 00009A13 A801                <1>       test   al, 01h
6955 00009A15 7406                <1>       jz     short pass_put_attr_r
6956 00009A17 66C7065200          <1>       mov    word [esi], 0052h     ; R
6957 00009A1C 46                  <1>       inc    esi
6958                              <1>
6959                              <1> pass_put_attr_r:
6960 00009A1D 3C20                <1>       cmp    al, 20h
6961 00009A1F 7205                <1>       jb     short pass_put_attr_a
6962 00009A21 66C7064100          <1>       mov    word [esi], 0041h     ; A
6963                              <1>
6964                              <1> pass_put_attr_a:
6965 00009A26 BE[693E0100]        <1>       mov    esi, Str_Attributes
6966 00009A2B E815D6FFFF          <1>       call   print_msg
6967 00009A30 BE[D7430100]        <1>       mov    esi, nextline
6968 00009A35 E80BD6FFFF          <1>       call   print_msg
6969 00009A3A E944F9FFFF          <1>       jmp    loc_file_rw_restore_retn
6970                              <1>
6971                              <1> loc_attr_file_check_fname_fchar:
6972 00009A3F 46                  <1>       inc    esi
6973 00009A40 803E20              <1>       cmp    byte [esi], 20h
6974 00009A43 74FA                <1>       je     short loc_attr_file_check_fname_fchar
6975 00009A45 0F8275FFFFFF        <1>       jb    pass_change_attr
6976                              <1>
6977                              <1> loc_attr_file_parse_path_name:
6978 00009A4B BF[0E8A0100]        <1>       mov    edi, FindFile_Drv
6979 00009A50 E8D1160000          <1>       call   parse_path_name
6980 00009A55 0F82EAEFFFFF        <1>       jc     loc_cmd_failed
6981                              <1>
6982                              <1> loc_attr_file_check_filename_exists:
6983 00009A5B BE[508A0100]        <1>       mov    esi, FindFile_Name
6984 00009A60 803E20              <1>       cmp    byte [esi], 20h
6985 00009A63 0F86DCEFFFFF        <1>       jna    loc_cmd_failed
6986 00009A69 8935[CC8A0100]      <1>       mov    [DelFile_FNPointer], esi
6987                              <1>
6988                              <1> loc_attr_file_drv:
6989 00009A6F 8A35[6E810100]      <1>       mov    dh, [Current_Drv]
6990 00009A75 8835[CA880100]      <1>       mov    [RUN_CDRV], dh
6991                              <1>
6992 00009A7B 8A15[0E8A0100]      <1>       mov    dl, [FindFile_Drv]
6993 00009A81 38F2                <1>       cmp    dl, dh
6994 00009A83 740B                <1>       je     short loc_attr_file_change_directory
6995                              <1>
6996 00009A85 E828E1FFFF          <1>       call   change_current_drive
6997 00009A8A 0F82F3F8FFFF        <1>       jc     loc_file_rw_cmd_failed
6998                              <1>
6999                              <1> loc_attr_file_change_directory:
7000 00009A90 803D[0F8A0100]20    <1>         cmp     byte [FindFile_Directory], 20h
7001 00009A97 7618                <1>       jna    short loc_attr_file_find
7002                              <1>
7003 00009A99 FE05[55390100]      <1>       inc    byte [Restore_CDIR]
7004                              <1>
7005 00009A9F BE[0F8A0100]        <1>       mov    esi, FindFile_Directory
7006 00009AA4 30E4                <1>       xor    ah, ah ; CD_COMMAND sign -> 0
7007 00009AA6 E865100000          <1>       call   change_current_directory
7008 00009AAB 0F82D2F8FFFF        <1>       jc     loc_file_rw_cmd_failed
7009                              <1>
7010                              <1> ;loc_attr_file_change_prompt_dir_string:
7011                              <1>       ;call  change_prompt_dir_string
7012                              <1>
7013                              <1> loc_attr_file_find:
7014                              <1>       ;mov   esi, FindFile_Name
7015 00009AB1 8B35[CC8A0100]      <1>       mov    esi, [DelFile_FNPointer]
7016 00009AB7 66B80008            <1>       mov    ax, 0800h ; Except volume labels
7017 00009ABB E895F4FFFF          <1>       call   find_first_file
7018 00009AC0 0F82BDF8FFFF        <1>       jc     loc_file_rw_cmd_failed
7019                              <1>
7020                              <1> loc_attr_file_ambgfn_check:
7021 00009AC6 6609D2              <1>       or     dx, dx ; Ambiguous filename chars used sign (DX>0)
7022                              <1>       ;      (Note: It was BX in TRDOS v1)
7023                              <1>       ;jz    short loc_attr_file_found
7024 00009AC9 0F85B8FDFFFF        <1>       jnz   loc_file_not_found ; 06/03/2016
7025                              <1>
7026                              <1>       ;mov   eax, 2 ; File not found sign
7027                              <1>       ;stc
7028                              <1>       ;jmp   loc_file_rw_cmd_failed
7029                              <1>
7030                              <1> loc_attr_file_found:
7031                              <1>       ; EDI = Directory buffer entry offset/address
7032                              <1>       ; BL = File (or Directory) Attributes
7033                              <1>       ;      (Note: It was 'CL' in TRDOS v1)
7034                              <1>       ; mov  bl, [EDI+0Bh]
7035                              <1>
7036 00009ACF 66833D[763E0100]00  <1>       cmp    word [Attr_Chars], 0
7037 00009AD7 770B                <1>       ja     short loc_attr_file_change_attributes
7038 00009AD9 881D[248B0100]      <1>       mov    [Attributes], bl
7039 00009ADF E9F4FEFFFF          <1>       jmp    loc_show_attributes
7040                              <1>
7041                              <1> loc_attr_file_change_attributes:
```

```
7042 00009AE4 A0[763E0100]        <1>       mov    al, [Attr_Chars]
7043 00009AE9 F6D0                <1>       not    al
7044 00009AEB 20C3                <1>       and    bl, al
7045 00009AED A0[773E0100]        <1>       mov    al, [Attr_Chars+1]
7046 00009AF2 08C3                <1>       or     bl, al
7047                              <1>
7048 00009AF4 66817F0CA101        <1>       cmp    word [edi+DirEntry_NTRes], 01A1h ; Singlix FS
7049 00009AFA 741D                <1>       je     short loc_attr_file_fs_check
7050                              <1>
7051 00009AFC 881D[248B0100]      <1>       mov    [Attributes], bl
7052 00009B02 885F0B              <1>       mov    [edi+0Bh], bl    ; Attributes (New!)
7053                              <1>
7054                              <1>       ; 04/03/2016
7055                              <1>       ; TRDOS v1 has a bug here! it does not set
7056                              <1>       ; 'DirBuff_ValidData' to 2; as result of this bug,
7057                              <1>       ; 'save_directory_buffer' would not save the new attributes !
7058                              <1>
7059 00009B05 C605[94880100]02    <1>       mov    byte [DirBuff_ValidData], 2
7060                              <1>
7061 00009B0C E8791A0000          <1>       call   save_directory_buffer
7062 00009B11 0F826CF8FFFF        <1>       jc     loc_file_rw_cmd_failed
7063                              <1>
7064 00009B17 EB33                <1>       jmp    short loc_print_attr_changed_message
7065                              <1>
7066                              <1> loc_attr_file_fs_check:
7067 00009B19 29C0                <1>       sub    eax, eax
7068 00009B1B 8A25[92880100]      <1>       mov    ah, [DirBuff_DRV]
7069 00009B21 BE00010900          <1>       mov    esi, Logical_DOSDisks
7070 00009B26 01C6                <1>       add    esi, eax
7071 00009B28 807E04A1            <1>       cmp    byte [esi+LD_FSType], 0A1h
7072 00009B2C 7309                <1>       jnc    short loc_attr_file_change_fs_file_attributes
7073                              <1>       ; 29/12/2017 (0Dh -> 29)
7074 00009B2E 66B81D00            <1>       mov    ax, 29 ; Invalid Data
7075 00009B32 E94CF8FFFF          <1>       jmp    loc_file_rw_cmd_failed
7076                              <1>
7077                              <1> loc_attr_file_change_fs_file_attributes:
7078                              <1>       ; BL = New MS-DOS File Attributes
7079 00009B37 88D8                <1>       mov    al, bl ; File/Directory Attributes
7080 00009B39 30E4                <1>       xor    ah, ah ; Attributes in MS-DOS format sign
7081 00009B3B E873050000          <1>       call   change_fs_file_attributes
7082 00009B40 0F823DF8FFFF        <1>       jc     loc_file_rw_cmd_failed
7083                              <1>
7084 00009B46 881D[248B0100]      <1>       mov    [Attributes], bl
7085                              <1>
7086                              <1> loc_print_attr_changed_message:
7087 00009B4C BE[643E0100]        <1>       mov    esi, Msg_New
7088 00009B51 E8EFD4FFFF          <1>       call   print_msg
7089 00009B56 E987FEFFFF          <1>       jmp    loc_show_attributes_no_nextline
7090                              <1>
7091                              <1> rename_file:
7092                              <1>       ; 13/11/2017
7093                              <1>       ; 06/11/2016
7094                              <1>       ; 05/11/2016
7095                              <1>       ; 16/10/2016
7096                              <1>       ; 08/03/2016
7097                              <1>       ; 06/03/2016 (TRDOS 386 = TRDOS v2.0)
7098                              <1>       ; 20/11/2010 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_rename')
7099                              <1>       ; 16/11/2010
7100                              <1>
7101                              <1> get_rename_source_fchar:
7102                              <1>       ; esi = file name
7103 00009B5B 803E20              <1>       cmp    byte [esi], 20h
7104 00009B5E 7614                <1>       jna    short loc_rename_nofilename_retn
7105                              <1>
7106 00009B60 8935[4C8B0100]      <1>       mov    [SourceFilePath], esi
7107                              <1>
7108                              <1> rename_scan_source_file:
7109 00009B66 46                  <1>       inc    esi
7110 00009B67 803E20              <1>       cmp    byte [esi], 20h
7111 00009B6A 7409                <1>       je     short rename_scan_destination_file_1
7112                              <1>       ;jb    short loc_rename_nofilename_retn
7113 00009B6C 0F82D3EEFFFF        <1>       jb     loc_cmd_failed
7114 00009B72 EBF2                <1>       jmp    short rename_scan_source_file
7115                              <1>
7116                              <1> loc_rename_nofilename_retn: ; 08/03/2016
7117 00009B74 C3                  <1>       retn
7118                              <1>
7119                              <1> rename_scan_destination_file_1:
7120 00009B75 C60600              <1>       mov    byte [esi], 0
7121                              <1>
7122                              <1> rename_scan_destination_file_2:
7123 00009B78 46                  <1>       inc    esi
7124 00009B79 803E20              <1>       cmp    byte [esi], 20h
7125 00009B7C 74FA                <1>       je     short rename_scan_destination_file_2
7126                              <1>       ;jb    short loc_rename_nofilename_retn
7127 00009B7E 0F82C1EEFFFF        <1>       jb     loc_cmd_failed
7128                              <1>
7129 00009B84 8935[508B0100]      <1>       mov    [DestinationFilePath], esi
7130                              <1>
7131                              <1> rename_scan_destination_file_3:
7132 00009B8A 46                  <1>       inc    esi
7133 00009B8B 803E20              <1>       cmp    byte [esi], 20h
7134 00009B8E 77FA                <1>       ja     short rename_scan_destination_file_3
7135                              <1>
7136 00009B90 C60600              <1>       mov    byte [esi], 0
7137                              <1>
7138                              <1> loc_rename_save_current_drive:
7139 00009B93 8A35[6E810100]      <1>       mov    dh, [Current_Drv]
7140 00009B99 8835[CA880100]      <1>       mov    byte [RUN_CDRV], dh
7141                              <1>
7142                              <1> loc_rename_sf_parse_path_name:
7143 00009B9F 8B35[4C8B0100]      <1>       mov    esi, [SourceFilePath]
7144 00009BA5 BF[0E8A0100]        <1>       mov    edi, FindFile_Drv
7145 00009BAA E877150000          <1>       call   parse_path_name
7146 00009BAF 0F8290EEFFFF        <1>       jc     loc_cmd_failed
```

```
7147                                    <1>
7148                                    <1> loc_rename_sf_check_filename_exists:
7149 00009BB5 BE[508A0100]             <1>         mov    esi, FindFile_Name
7150 00009BBA 803E20                   <1>         cmp    byte [esi], 20h
7151 00009BBD 0F8682EEFFFF             <1>         jna    loc_cmd_failed
7152                                    <1>
7153                                    <1>         ;mov   [DelFile_FNPointer], esi
7154                                    <1>
7155                                    <1> loc_rename_sf_drv:
7156                                    <1>         ;mov   dh, [Current_Drv]
7157                                    <1>         ;mov   [RUN_CDRV], dh
7158                                    <1>
7159 00009BC3 8A15[0E8A0100]           <1>         mov    dl, [FindFile_Drv]
7160 00009BC9 38F2                     <1>         cmp    dl, dh ; dh = [Current_Drv]
7161 00009BCB 740B                     <1>         je     short rename_sf_change_directory
7162                                    <1>
7163 00009BCD E8E0DFFFFF               <1>         call   change_current_drive
7164 00009BD2 0F82ABF7FFFF             <1>         jc     loc_file_rw_cmd_failed
7165                                    <1>
7166                                    <1> rename_sf_change_directory:
7167 00009BD8 803D[0F8A0100]20         <1>         cmp    byte [FindFile_Directory], 20h
7168 00009BDF 7618                     <1>         jna    short rename_sf_find
7169                                    <1>
7170 00009BE1 FE05[55390100]           <1>         inc    byte [Restore_CDIR]
7171 00009BE7 BE[0F8A0100]             <1>         mov    esi, FindFile_Directory
7172 00009BEC 30E4                     <1>         xor    ah, ah ; CD_COMMAND sign -> 0
7173 00009BEE E81D0F0000               <1>         call   change_current_directory
7174 00009BF3 0F828AF7FFFF             <1>         jc     loc_file_rw_cmd_failed
7175                                    <1>
7176                                    <1> ;rename_sf_change_prompt_dir_string:
7177                                    <1>         ;call  change_prompt_dir_string
7178                                    <1>
7179                                    <1> rename_sf_find:
7180                                    <1>         ;mov   esi, [DelFile_FNPointer]
7181 00009BF9 BE[508A0100]             <1>         mov    esi, FindFile_Name
7182                                    <1>
7183 00009BFE 66B80008                 <1>         mov    ax, 0800h ; Except volume labels
7184 00009C02 E84EF3FFFF               <1>         call   find_first_file
7185 00009C07 0F8276F7FFFF             <1>         jc     loc_file_rw_cmd_failed
7186                                    <1>
7187                                    <1> loc_rename_sf_ambgfn_check:
7188 00009C0D 6621D2                   <1>         and    dx, dx ; Ambiguous filename chars used sign (DX>0)
7189                                    <1>         ;      (Note: It was BX in TRDOS v1)
7190                                    <1>         ;jz    short loc_rename_sf_found
7191 00009C10 0F8571FCFFFF             <1>         jnz    loc_file_not_found
7192                                    <1>
7193                                    <1>         ;mov   eax, 2 ; File not found sign
7194                                    <1>         ;stc
7195                                    <1>         ;jmp   loc_file_rw_cmd_failed
7196                                    <1>
7197                                    <1> loc_rename_sf_found:
7198                                    <1>         ; EDI = Directory buffer entry offset/address
7199                                    <1>         ; BL = File (or Directory) Attributes
7200                                    <1>         ;      (Note: It was 'CL' in TRDOS v1)
7201                                    <1>         ; mov  bl, [EDI+0Bh]
7202                                    <1>
7203 00009C16 F6C307                   <1>         test   bl, 07h ; Attributes, S-H-R
7204 00009C19 0F856EF7FFFF             <1>         jnz    loc_permission_denied
7205                                    <1>
7206 00009C1F BE[0E8A0100]             <1>         mov   esi, FindFile_Drv
7207 00009C24 BF[548B0100]             <1>         mov   edi, SourceFile_Drv
7208 00009C29 B920000000               <1>         mov    ecx, 32
7209 00009C2E F3A5                     <1>         rep    movsd
7210                                    <1>
7211                                    <1> loc_rename_df_parse_path_name:
7212 00009C30 8B35[508B0100]           <1>         mov    esi, [DestinationFilePath]
7213 00009C36 BF[0E8A0100]             <1>         mov    edi, FindFile_Drv
7214 00009C3B E8E6140000               <1>         call   parse_path_name
7215 00009C40 7219                     <1>         jc     short loc_rename_df_cmd_failed
7216                                    <1>
7217                                    <1>         ;mov   dh, [RUN_CDRV]
7218 00009C42 8A35[6E810100]           <1>         mov    dh, [Current_Drv]
7219                                    <1>
7220                                    <1>         ; 'rename' command is valid only for same dos drive and same dir!
7221                                    <1>         ; ('move' command must be used if source file and destination file
7222                                    <1>         ; directories are not same!)
7223 00009C48 8A15[0E8A0100]           <1>         mov    dl, [FindFile_Drv]
7224 00009C4E 38F2                     <1>         cmp    dl, dh ; are source and destination drives different ?!
7225 00009C50 7509                     <1>         jne    short loc_rename_df_cmd_failed ; yes!
7226                                    <1>
7227                                    <1> rename_df_check_dirname_exists:
7228 00009C52 803D[0F8A0100]00         <1>         cmp    byte [FindFile_Directory], 0
7229 00009C59 760B                     <1>         jna    short rename_df_check_filename_exists
7230                                    <1>
7231                                    <1>         ; different source file and destination file directories !
7232                                    <1> loc_rename_df_cmd_failed:
7233 00009C5B B801000000               <1>         mov    eax, 1 ; TRDOS 'Bad command or file name' error
7234 00009C60 F9                       <1>         stc
7235 00009C61 E91DF7FFFF               <1>         jmp    loc_file_rw_cmd_failed
7236                                    <1>
7237                                    <1> rename_df_check_filename_exists:
7238 00009C66 BE[508A0100]             <1>         mov    esi, FindFile_Name
7239 00009C6B E8A3F6FFFF               <1>         call   check_filename
7240 00009C70 0F82BFF7FFFF             <1>         jc     loc_mkdir_invalid_dir_name_chars
7241                                    <1>
7242                                    <1>         ;mov   [DelFile_FNPointer], esi
7243                                    <1>         ;cmp   byte [esi], 20h
7244                                    <1>         ;ja    short loc_rename_df_find
7245                                    <1>
7246                                    <1>         ;mov   dh, [Current_Drv] ; dh has not been changed
7247                                    <1>
7248                                    <1> rename_df_drv_check_writable:
7249 00009C76 0FB6F6                   <1>         movzx  esi, dh
7250                                    <1>         ;movzx esi, byte [Current_Drv]
7251 00009C79 81C600010900             <1>         add    esi, Logical_DOSDisks
```

```
7252                            <1>
7253 00009C7F 88F2              <1>       mov    dl, dh ; dl = [Current_Drv]
7254 00009C81 8A7601            <1>       mov    dh, [esi+LD_DiskType]
7255                            <1>
7256 00009C84 80FE01            <1>       cmp    dh, 1 ; 0 = Invalid
7257 00009C87 7310              <1>       jnb    short rename_df_compare_sf_df_name
7258                            <1>
7259                            <1>       ; 16/10/2016 (13h -> 30)
7260 00009C89 B81E000000        <1>       mov    eax, 30 ; 'Disk write-protected' error
7261 00009C8E 8B1D[508B0100]    <1>       mov    ebx, [DestinationFilePath]
7262 00009C94 E9EAF6FFFF        <1>       jmp    loc_file_rw_cmd_failed
7263                            <1>
7264                            <1> rename_df_compare_sf_df_name:
7265 00009C99 BE[508A0100]      <1>       mov    esi, FindFile_Name
7266 00009C9E BF[968B0100]      <1>       mov    edi, SourceFile_Name
7267 00009CA3 B90C000000        <1>       mov    ecx, 12
7268                            <1> rename_df_compare_sf_df_name_next:
7269 00009CA8 AC                <1>       lodsb
7270 00009CA9 AE                <1>       scasb
7271 00009CAA 7506              <1>       jne    short loc_rename_df_find
7272 00009CAC 08C0              <1>       or     al, al
7273 00009CAE 74AB              <1>       jz     short loc_rename_df_cmd_failed
7274 00009CB0 E2F6              <1>       loop   rename_df_compare_sf_df_name_next
7275                            <1>
7276                            <1> loc_rename_df_find:
7277                            <1>       ;mov   esi, [DelFile_FNPointer]
7278 00009CB2 BE[508A0100]      <1>       mov    esi, FindFile_Name
7279                            <1>
7280 00009CB7 6631C0            <1>       xor    ax, ax ; Any
7281 00009CBA E896F2FFFF        <1>       call   find_first_file
7282                            <1>       ;jnc   short loc_rename_df_found
7283                            <1>       ; 29/12/2017
7284 00009CBF 0F83C8F6FFFF      <1>       jnc    loc_permission_denied
7285                            <1>
7286                            <1> loc_rename_df_check_error_code:
7287                            <1>       ;cmp   eax, 2
7288 00009CC5 3C02              <1>       cmp    al, 2 ; Not found error
7289 00009CC7 7406              <1>       je     short rename_df_move_find_struct_to_dest
7290 00009CC9 F9                <1>       stc
7291 00009CCA E9B4F6FFFF        <1>       jmp    loc_file_rw_cmd_failed
7292                            <1>
7293                            <1> ;loc_rename_df_found:
7294                            <1>       ; 05/11/2016
7295                            <1>       ; Permission denied error
7296                            <1>       ;mov   eax, ERR_PERM_DENIED ; 29/12/2017
7297                            <1>       ;stc
7298                            <1>       ;jmp   loc_permission_denied  ; 06/11/2016
7299                            <1>
7300                            <1> rename_df_move_find_struct_to_dest:
7301 00009CCF BE[0E8A0100]      <1>       mov   esi, FindFile_Drv
7302 00009CD4 BF[D48B0100]      <1>       mov   edi, DestinationFile_Drv
7303 00009CD9 B920000000        <1>       mov    ecx, 32
7304 00009CDE F3A5              <1>       rep    movsd
7305                            <1>
7306                            <1> loc_rename_df_process_q_sf:
7307                            <1>       ;mov   ecx, 12
7308 00009CE0 B10C              <1>       mov    cl, 12
7309 00009CE2 BE[968B0100]      <1>       mov    esi, SourceFile_Name
7310 00009CE7 BF[A53E0100]      <1>       mov    edi, Rename_OldName
7311                            <1> rename_df_process_q_nml_1_sf:
7312 00009CEC AC                <1>       lodsb
7313 00009CED 3C20              <1>       cmp    al, 20h
7314 00009CEF 7603              <1>       jna    short rename_df_process_q_nml_2_sf
7315 00009CF1 AA                <1>       stosb
7316 00009CF2 E2F8              <1>       loop   rename_df_process_q_nml_1_sf
7317                            <1>
7318                            <1> rename_df_process_q_nml_2_sf:
7319 00009CF4 C60700            <1>       mov    byte [edi], 0
7320                            <1>
7321                            <1> loc_rename_df_process_q_df:
7322                            <1>       ;mov   ecx, 12
7323 00009CF7 B10C              <1>       mov    cl, 12
7324 00009CF9 BE[168C0100]      <1>       mov    esi, DestinationFile_Name
7325 00009CFE BF[B63E0100]      <1>       mov    edi, Rename_NewName
7326                            <1> rename_df_process_q_nml_1_df:
7327 00009D03 AC                <1>       lodsb
7328 00009D04 3C20              <1>       cmp    al, 20h
7329 00009D06 7603              <1>       jna    short loc_rename_df_process_q_nml_2_df
7330 00009D08 AA                <1>       stosb
7331 00009D09 E2F8              <1>       loop   rename_df_process_q_nml_1_df
7332                            <1>
7333                            <1> loc_rename_df_process_q_nml_2_df:
7334 00009D0B C60700            <1>       mov    byte [edi], 0
7335                            <1>
7336                            <1> loc_rename_confirmation_question:
7337 00009D0E BE[7D3E0100]      <1>       mov    esi, Msg_DoYouWantRename
7338 00009D13 E82DD3FFFF        <1>       call   print_msg
7339                            <1>
7340 00009D18 A0[B18B0100]      <1>       mov    al, [SourceFile_DirEntry+11] ; Attributes
7341 00009D1D 2410              <1>       and    al, 10h
7342 00009D1F 750C              <1>       jnz    short rename_confirmation_question_dir
7343                            <1>
7344                            <1> rename_confirmation_question_file:
7345 00009D21 BE[943E0100]      <1>       mov    esi, Rename_File
7346 00009D26 E81AD3FFFF        <1>       call   print_msg
7347 00009D2B EB0A              <1>       jmp    short rename_confirmation_question_as
7348                            <1>
7349                            <1> rename_confirmation_question_dir:
7350 00009D2D BE[9A3E0100]      <1>       mov    esi, Rename_Directory
7351 00009D32 E80ED3FFFF        <1>       call   print_msg
7352                            <1>
7353                            <1> rename_confirmation_question_as:
7354 00009D37 BE[A53E0100]      <1>       mov    esi, Rename_OldName
7355 00009D3C E804D3FFFF        <1>       call   print_msg
7356 00009D41 BE[B23E0100]      <1>       mov    esi, Msg_File_rename_as
```

```
7357 00009D46 E8FAD2FFFF          <1>       call   print_msg
7358 00009D4B BE[D93D0100]        <1>       mov    esi, Msg_YesNo
7359 00009D50 E8F0D2FFFF          <1>       call   print_msg
7360                              <1>
7361                              <1> loc_rename_ask_again:
7362 00009D55 30E4                <1>       xor    ah, ah
7363 00009D57 E87F71FFFF          <1>       call   int16h
7364 00009D5C 3C1B                <1>       cmp    al, 1Bh
7365 00009D5E 740F                <1>       je     short loc_do_not_rename_file
7366 00009D60 24DF                <1>       and    al, 0DFh
7367 00009D62 A2[E33D0100]        <1>       mov    [Y_N_nextline], al
7368 00009D67 3C59                <1>       cmp    al, 'Y'
7369 00009D69 7404                <1>       je     short loc_yes_rename_file
7370 00009D6B 3C4E                <1>       cmp    al, 'N'
7371 00009D6D 75E6                <1>       jne    short loc_rename_ask_again
7372                              <1>
7373                              <1> loc_do_not_rename_file:
7374                              <1> loc_yes_rename_file:
7375 00009D6F E848F7FFFF          <1>       call   y_n_answer ; 29/12/2017
7376                              <1>       ;cmp   al, 'Y' ; 'yes'
7377                              <1>       ;cmc
7378                              <1>       ;jnc   loc_file_rw_restore_retn
7379 00009D74 3C4E                <1>       cmp    al, 'N' ; 'no'
7380 00009D76 0F8407F6FFFF        <1>       je     loc_file_rw_restore_retn
7381                              <1>
7382 00009D7C BE[B63E0100]        <1>       mov    esi, Rename_NewName
7383 00009D81 668B0D[CE8B0100]    <1>       mov    cx, [SourceFile_DirEntryNumber]
7384 00009D88 66A1[BA8B0100]      <1>       mov    ax, [SourceFile_DirEntry+20] ; First Cluster, HW
7385 00009D8E C1E010              <1>       shl    eax, 16 ; 13/11/2017
7386 00009D91 66A1[C08B0100]      <1>       mov    ax, [SourceFile_DirEntry+26] ; First Cluster, LW
7387                              <1>
7388 00009D97 0FB61D[A38B0100]    <1>       movzx  ebx, byte [SourceFile_LongNameEntryLength]
7389 00009D9E E8CB1B0000          <1>       call   rename_directory_entry
7390 00009DA3 E9FBF6FFFF          <1>       jmp    loc_rename_file_ok
7391                              <1> ;loc_rename_file_ok:
7392                              <1> ;     jc     loc_run_cmd_failed
7393                              <1> ;     mov    esi, Msg_OK
7394                              <1> ;     call   proc_printmsg
7395                              <1> ;     jmp    loc_file_rw_restore_retn
7396                              <1>
7397                              <1> move_file:
7398                              <1>       ; 11/03/2016
7399                              <1>       ; 09/03/2016
7400                              <1>       ; 08/03/2016 (TRDOS 386 = TRDOS v2.0)
7401                              <1>       ; 21/05/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_move')
7402                              <1>       ; 23/04/2011
7403                              <1>
7404                              <1> get_move_source_fchar:
7405                              <1>       ; esi = file name
7406 00009DA8 803E20              <1>       cmp    byte [esi], 20h
7407 00009DAB 7614                <1>       jna    short loc_move_nofilename_retn
7408                              <1>
7409 00009DAD 8935[4C8B0100]      <1>       mov    [SourceFilePath], esi
7410                              <1>
7411                              <1> move_scan_source_file:
7412 00009DB3 46                  <1>       inc    esi
7413 00009DB4 803E20              <1>       cmp    byte [esi], 20h
7414 00009DB7 7409                <1>       je    short move_scan_destination_1
7415                              <1>       ;jb    short loc_move_nofilename_retn
7416 00009DB9 0F8286ECFFFF        <1>       jb     loc_cmd_failed
7417 00009DBF EBF2                <1>       jmp    short move_scan_source_file
7418                              <1>
7419                              <1> loc_move_nofilename_retn:
7420 00009DC1 C3                  <1>       retn
7421                              <1>
7422                              <1> move_scan_destination_1:
7423 00009DC2 C60600              <1>       mov    byte [esi], 0
7424                              <1>
7425                              <1> move_scan_destination_2:
7426 00009DC5 46                  <1>       inc    esi
7427 00009DC6 803E20              <1>       cmp    byte [esi], 20h
7428 00009DC9 74FA                <1>       je     short move_scan_destination_2
7429                              <1>       ;jb    short loc_move_nofilename_retn
7430 00009DCB 0F8274ECFFFF        <1>       jb     loc_cmd_failed
7431                              <1>
7432 00009DD1 8935[508B0100]      <1>       mov    [DestinationFilePath], esi
7433                              <1>
7434                              <1> move_scan_destination_3:
7435 00009DD7 46                  <1>       inc    esi
7436 00009DD8 803E20              <1>       cmp    byte [esi], 20h
7437 00009DDB 77FA                <1>       ja     short move_scan_destination_3
7438 00009DDD C60600              <1>       mov    byte [esi], 0
7439                              <1>
7440                              <1> loc_move_scan_destination_OK:
7441 00009DE0 8B35[4C8B0100]      <1>       mov    esi, [SourceFilePath]
7442 00009DE6 8B3D[508B0100]      <1>       mov    edi, [DestinationFilePath]
7443                              <1>
7444 00009DEC B001                <1>       mov    al, 1  ; move procedure Phase 1
7445 00009DEE E8F71B0000          <1>       call   move_source_file_to_destination_file
7446 00009DF3 7328                <1>       jnc    short move_source_file_to_destination_question
7447                              <1>
7448                              <1> loc_move_cmd_failed_1:
7449 00009DF5 08C0                <1>       or     al, al
7450 00009DF7 0F8448ECFFFF        <1>       jz     loc_cmd_failed
7451 00009DFD 3C11                <1>       cmp    al, 11h
7452 00009DFF 740D                <1>       je     short loc_msg_not_same_device
7453                              <1>       ;cmp   al, 05h
7454                              <1>       ;cmp   al, ERR_PERM_DENIED ; 29/12/2017
7455                              <1>       ;jne   loc_run_cmd_failed
7456                              <1>       ;jmp   loc_permission_denied
7457 00009E01 3C0B                <1>       cmp    al, ERR_PERM_DENIED
7458 00009E03 0F8484F5FFFF        <1>       je     loc_permission_denied
7459 00009E09 E962ECFFFF          <1>       jmp    loc_run_cmd_failed
7460                              <1>
7461                              <1>       ;mov   esi, Msg_Permission_denied
```

```
7462                            <1>       ;call  print_msg
7463                            <1>       ;jmp   loc_file_rw_restore_retn
7464                            <1>
7465                            <1> loc_msg_not_same_device:
7466 00009E0E BE[C33E0100]     <1>       mov    esi, msg_not_same_drv
7467 00009E13 E82DD2FFFF       <1>       call   print_msg
7468 00009E18 E966F5FFFF       <1>       jmp    loc_file_rw_restore_retn
7469                            <1>
7470                            <1> move_source_file_to_destination_question:
7471 00009E1D A0[548B0100]     <1>       mov   al, [SourceFile_Drv]
7472 00009E22 0441             <1>       add    al, 'A'
7473 00009E24 A2[253F0100]     <1>       mov    [msg_source_file_drv], al
7474 00009E29 A0[D48B0100]     <1>       mov   al, [DestinationFile_Drv]
7475 00009E2E 0441             <1>       add    al, 'A'
7476 00009E30 A2[443F0100]     <1>       mov    [msg_destination_file_drv], al
7477                            <1>
7478 00009E35 57               <1>       push   edi ; *
7479                            <1>
7480 00009E36 BE[093F0100]     <1>       mov    esi, msg_source_file
7481 00009E3B E805D2FFFF       <1>       call   print_msg
7482 00009E40 BE[558B0100]     <1>       mov    esi, SourceFile_Directory
7483 00009E45 803E20           <1>       cmp    byte [esi], 20h
7484 00009E48 7605             <1>       jna    short msftdfq_sfn
7485 00009E4A E8F6D1FFFF       <1>       call   print_msg
7486                            <1> msftdfq_sfn:
7487 00009E4F BE[968B0100]     <1>       mov    esi, SourceFile_Name
7488 00009E54 E8ECD1FFFF       <1>       call   print_msg
7489 00009E59 BE[283F0100]     <1>       mov    esi, msg_destination_file
7490 00009E5E E8E2D1FFFF       <1>       call   print_msg
7491 00009E63 BE[D58B0100]     <1>       mov    esi, DestinationFile_Directory
7492 00009E68 803E20           <1>       cmp    byte [esi], 20h
7493 00009E6B 7605             <1>       jna    short msftdfq_dfn
7494 00009E6D E8D3D1FFFF       <1>       call   print_msg
7495                            <1> msftdfq_dfn:
7496 00009E72 BE[168C0100]     <1>       mov    esi, DestinationFile_Name
7497 00009E77 E8C9D1FFFF       <1>       call   print_msg
7498 00009E7C BE[473F0100]     <1>       mov    esi, msg_copy_nextline
7499 00009E81 E8BFD1FFFF       <1>       call   print_msg
7500 00009E86 BE[473F0100]     <1>       mov    esi, msg_copy_nextline
7501 00009E8B E8B5D1FFFF       <1>       call   print_msg
7502                            <1>
7503                            <1> loc_move_ask_for_new_file_yes_no:
7504 00009E90 BE[D53E0100]     <1>       mov    esi, Msg_DoYouWantMoveFile
7505 00009E95 E8ABD1FFFF       <1>       call   print_msg
7506 00009E9A BE[D93D0100]     <1>       mov    esi, Msg_YesNo
7507 00009E9F E8A1D1FFFF       <1>       call   print_msg
7508                            <1> loc_move_ask_for_new_file_again:
7509 00009EA4 30E4             <1>       xor    ah, ah
7510 00009EA6 E83070FFFF       <1>       call   int16h
7511 00009EAB 3C1B             <1>       cmp    al, 1Bh
7512                            <1>       ;je    short loc_do_not_move_file
7513 00009EAD 7441             <1>       je     short loc_move_y_n_escape
7514 00009EAF 24DF             <1>       and    al, 0DFh
7515 00009EB1 A2[E33D0100]     <1>       mov   [Y_N_nextline], al
7516 00009EB6 3C59             <1>       cmp    al, 'Y'
7517 00009EB8 7404             <1>       je     short loc_yes_move_file
7518 00009EBA 3C4E             <1>       cmp    al, 'N'
7519 00009EBC 75E6             <1>       jne    short loc_move_ask_for_new_file_again
7520                            <1>
7521                            <1> loc_do_not_move_file:
7522                            <1> loc_yes_move_file:
7523 00009EBE E8F9F5FFFF       <1>       call   y_n_answer ; 29/12/2017
7524 00009EC3 5F               <1>       pop    edi ; *
7525                            <1>       ;cmp   al, 'Y' ; 'yes'
7526                            <1>       ;cmc
7527                            <1>       ;jnc   loc_file_rw_restore_retn
7528 00009EC4 3C4E             <1>       cmp    al, 'N' ; 'no'
7529 00009EC6 0F84B7F4FFFF     <1>       je     loc_file_rw_restore_retn
7530                            <1>
7531                            <1> loc_move_yes_move_file:
7532 00009ECC B002             <1>       mov    al, 2 ; move procedure Phase 2
7533 00009ECE E8171B0000       <1>       call   move source file to destination file
7534                            <1>       ;jc    short loc_move_cmd_failed_2
7535 00009ED3 0F83D0F5FFFF     <1>       jnc   move_source_file_to_destination_OK
7536                            <1>
7537                            <1> ;move_source_file_to_destination_OK:
7538                            <1> ;     mov    esi, Msg_OK
7539                            <1> ;     call   print_msg
7540                            <1> ;     jmp    loc_file_rw_restore_retn
7541                            <1>
7542                            <1> loc_move_cmd_failed_2:
7543 00009ED9 3C27             <1>       cmp    al, 27h
7544 00009EDB 0F858FEBFFFF     <1>       jne    loc_run_cmd_failed
7545                            <1>
7546 00009EE1 BE[EE3E0100]     <1>       mov    esi, msg_insufficient_disk_space
7547 00009EE6 E85AD1FFFF       <1>       call   print_msg
7548                            <1>
7549 00009EEB E993F4FFFF       <1>       jmp    loc_file_rw_restore_retn
7550                            <1>
7551                            <1> loc_move_y_n_escape:
7552 00009EF0 B04E             <1>       mov    al, 'N' ; 'no'
7553 00009EF2 EBCA             <1>       jmp    short loc_do_not_move_file
7554                            <1>
7555                            <1> copy_file:
7556                            <1>       ; 15/10/2016
7557                            <1>       ; 24/03/2016
7558                            <1>       ; 21/03/2016
7559                            <1>       ; 15/03/2016 (TRDOS 386 = TRDOS v2.0)
7560                            <1>       ; 21/05/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_copy')
7561                            <1>       ; 01/08/2010
7562                            <1>
7563                            <1> get_copy_source_fchar:
7564                            <1>       ; esi = file name
7565 00009EF4 803E20           <1>       cmp    byte [esi], 20h
7566 00009EF7 7614             <1>       jna    short loc_copy_nofilename_retn
```

```
7567                                    <1>
7568 00009EF9 8935[4C8B0100]            <1>         mov    [SourceFilePath], esi
7569                                    <1>
7570                                    <1> copy_scan_source_file:
7571 00009EFF 46                        <1>         inc    esi
7572 00009F00 803E20                    <1>         cmp    byte [esi], 20h
7573 00009F03 7409                      <1>         je     short copy_scan_destination_1
7574                                    <1>         ;jb    short loc_copy_nofilename_retn
7575 00009F05 0F823AEBFFFF              <1>         jb     loc_cmd_failed
7576 00009F0B EBF2                      <1>         jmp    short copy_scan_source_file
7577                                    <1>
7578                                    <1> loc_copy_nofilename_retn:
7579 00009F0D C3                        <1>         retn
7580                                    <1>
7581                                    <1> copy_scan_destination_1:
7582 00009F0E C60600                    <1>         mov    byte [esi], 0
7583                                    <1>
7584                                    <1> copy_scan_destination_2:
7585 00009F11 46                        <1>         inc    esi
7586 00009F12 803E20                    <1>         cmp    byte [esi], 20h
7587 00009F15 74FA                      <1>         je     short copy_scan_destination_2
7588                                    <1>         ;jb    short loc_copy_nofilename_retn
7589 00009F17 0F8228EBFFFF              <1>         jb     loc_cmd_failed
7590                                    <1>
7591 00009F1D 8935[508B0100]            <1>         mov    [DestinationFilePath], esi
7592                                    <1>
7593                                    <1> copy_scan_destination_3:
7594 00009F23 46                        <1>         inc    esi
7595 00009F24 803E20                    <1>         cmp    byte [esi], 20h
7596 00009F27 77FA                      <1>         ja     short copy_scan_destination_3
7597 00009F29 C60600                    <1>         mov    byte [esi], 0
7598                                    <1>
7599                                    <1> loc_copy_save_current_drive:
7600 00009F2C 8A35[6E810100]            <1>         mov    dh, [Current_Drv]
7601 00009F32 8835[CA880100]            <1>         mov    [RUN_CDRV], dh
7602                                    <1>
7603                                    <1> copy_source_file_to_destination_phase_1:
7604 00009F38 8B35[4C8B0100]            <1>         mov    esi, [SourceFilePath]
7605 00009F3E 8B3D[508B0100]            <1>         mov    edi, [DestinationFilePath]
7606                                    <1>
7607 00009F44 B001                      <1>         mov    al, 1  ; copy procedure Phase 1
7608 00009F46 E83C1D0000                <1>         call   copy_source_file_to_destination_file
7609 00009F4B 732B                      <1>         jnc    short copy_source_file_to_destination_question
7610                                    <1>
7611                                    <1> loc_copy_cmd_failed_1:
7612                                    <1>         ; 18/03/2016 (restore current drive and directory)
7613 00009F4D 08C0                      <1>         or     al, al
7614 00009F4F 7507                      <1>         jnz    short loc_copy_cmd_failed_2
7615                                    <1>
7616 00009F51 FEC0                      <1>         inc   al ; mov al, 1 ; Bad command or file name !
7617 00009F53 E918EBFFFF                <1>         jmp    loc_run_cmd_failed
7618                                    <1>
7619                                    <1> loc_copy_cmd_failed_2:
7620 00009F58 3C27                      <1>         cmp    al, 27h ; Insufficient disk space
7621 00009F5A 740D                      <1>         je     short loc_file_write_insuff_disk_space_msg
7622                                    <1>
7623                                    <1>         ; 29/12/2017
7624                                    <1>         ;cmp   al, 05h
7625 00009F5C 3C0B                      <1>         cmp    al, ERR_PERM_DENIED
7626 00009F5E 0F850CEBFFFF              <1>         jne    loc_run_cmd_failed
7627                                    <1>
7628 00009F64 E924F4FFFF                <1>         jmp    loc_permission_denied
7629                                    <1>
7630                                    <1> loc_file_write_insuff_disk_space_msg:
7631 00009F69 BE[EE3E0100]              <1>         mov    esi, msg_insufficient_disk_space
7632 00009F6E E8D2D0FFFF                <1>         call   print_msg
7633 00009F73 E90BF4FFFF                <1>         jmp   loc_file_rw_restore_retn
7634                                    <1>
7635                                    <1> copy_source_file_to_destination_question:
7636 00009F78 57                        <1>         push   edi ; *
7637                                    <1>
7638                                    <1>         ; dh = source file attributes
7639                                    <1>         ; dl > 0 -> destination file found
7640 00009F79 20D2                      <1>         and    dl, dl
7641 00009F7B 7449                      <1>         jz     short copy_source_file_to_destination_pass_owrq
7642                                    <1>
7643                                    <1> loc_copy_ask_for_owr_yes_no:
7644 00009F7D BE[4A3F0100]              <1>         mov    esi, Msg_DoYouWantOverWriteFile
7645 00009F82 E8BED0FFFF                <1>         call   print_msg
7646 00009F87 BE[168C0100]              <1>         mov    esi, DestinationFile_Name
7647 00009F8C E8B4D0FFFF                <1>         call   print_msg
7648 00009F91 BE[D93D0100]              <1>         mov    esi, Msg_YesNo
7649 00009F96 E8AAD0FFFF                <1>         call   print_msg
7650                                    <1>
7651                                    <1> loc_copy_ask_for_owr_again:
7652 00009F9B 30E4                      <1>         xor    ah, ah
7653 00009F9D E8396FFFFF                <1>         call   int16h
7654 00009FA2 3C1B                      <1>         cmp    al, 1Bh
7655                                    <1>         ;je   loc_do_not_copy_file
7656 00009FA4 7419                      <1>         je    short loc_copy_y_n_escape
7657 00009FA6 24DF                      <1>         and    al, 0DFh
7658 00009FA8 A2[E33D0100]              <1>         mov   [Y_N_nextline], al
7659 00009FAD 3C59                      <1>         cmp    al, 'Y'
7660 00009FAF 0F84B1000000              <1>         je    loc_yes_copy_file
7661 00009FB5 3C4E                      <1>         cmp    al, 'N'
7662 00009FB7 0F84A9000000              <1>         je    loc_do_not_copy_file
7663 00009FBD EBDC                      <1>         jmp    short loc_copy_ask_for_owr_again
7664                                    <1>
7665                                    <1> loc_copy_y_n_escape:
7666 00009FBF B04E                      <1>         mov    al, 'N' ; 'no'
7667 00009FC1 E9A0000000                <1>         jmp   loc_do_not_copy_file
7668                                    <1>
7669                                    <1> copy_source_file_to_destination_pass_owrq:
7670 00009FC6 A0[548B0100]              <1>         mov     al, [SourceFile_Drv]
7671 00009FCB 0441                      <1>         add    al, 'A'
```

```
7672 00009FCD A2[253F0100]        <1>       mov    [msg_source_file_drv], al
7673 00009FD2 A0[D48B0100]        <1>       mov   al, [DestinationFile_Drv]
7674 00009FD7 0441                <1>       add    al, 'A'
7675 00009FD9 A2[443F0100]        <1>       mov    [msg_destination_file_drv], al
7676                              <1>
7677 00009FDE BE[093F0100]        <1>       mov    esi, msg_source_file
7678 00009FE3 E85DD0FFFF          <1>       call   print_msg
7679 00009FE8 BE[558B0100]        <1>       mov    esi, SourceFile_Directory
7680 00009FED 803E20              <1>       cmp    byte [esi], 20h
7681 00009FF0 7605                <1>       jna    short csftdfq_sfn
7682 00009FF2 E84ED0FFFF          <1>       call   print_msg
7683                              <1> csftdfq_sfn:
7684 00009FF7 BE[968B0100]        <1>       mov    esi, SourceFile_Name
7685 00009FFC E844D0FFFF          <1>       call   print_msg
7686 0000A001 BE[283F0100]        <1>       mov    esi, msg_destination_file
7687 0000A006 E83AD0FFFF          <1>       call   print_msg
7688 0000A00B BE[D58B0100]        <1>       mov    esi, DestinationFile_Directory
7689 0000A010 803E20              <1>       cmp    byte [esi], 20h
7690 0000A013 7605                <1>       jna    short csftdfq_dfn
7691 0000A015 E82BD0FFFF          <1>       call   print_msg
7692                              <1> csftdfq_dfn:
7693 0000A01A BE[168C0100]        <1>       mov    esi, DestinationFile_Name
7694 0000A01F E821D0FFFF          <1>       call   print_msg
7695 0000A024 BE[473F0100]        <1>       mov    esi, msg_copy_nextline
7696 0000A029 E817D0FFFF          <1>       call   print_msg
7697 0000A02E BE[473F0100]        <1>       mov    esi, msg_copy_nextline
7698 0000A033 E80DD0FFFF          <1>       call   print_msg
7699                              <1>
7700                              <1> loc_copy_ask_for_new_file_yes_no:
7701 0000A038 BE[693F0100]        <1>       mov    esi, Msg_DoYouWantCopyFile
7702 0000A03D E803D0FFFF          <1>       call   print_msg
7703 0000A042 BE[D93D0100]        <1>       mov    esi, Msg_YesNo
7704 0000A047 E8F9CFFFFF          <1>       call   print_msg
7705                              <1>
7706                              <1> loc_copy_ask_for_new_file_again:
7707 0000A04C 30E4                <1>       xor    ah, ah
7708 0000A04E E8886EFFFF          <1>       call   int16h
7709 0000A053 3C1B                <1>       cmp    al, 1Bh
7710 0000A055 740F                <1>       je     short loc_do_not_copy_file
7711 0000A057 24DF                <1>       and    al, 0DFh
7712 0000A059 A2[E33D0100]        <1>       mov   [Y_N_nextline], al
7713 0000A05E 3C59                <1>       cmp    al, 'Y'
7714 0000A060 7404                <1>       je     short loc_yes_copy_file
7715 0000A062 3C4E                <1>       cmp    al, 'N'
7716 0000A064 75E6                <1>       jne    short loc_copy_ask_for_new_file_again
7717                              <1>
7718                              <1> loc_do_not_copy_file:
7719                              <1> loc_yes_copy_file:
7720 0000A066 E851F4FFFF          <1>       call   y_n_answer ; 29/12/2017
7721 0000A06B 5F                  <1>       pop    edi ; *
7722                              <1>       ;cmp   al, 'Y' ; 'yes'
7723                              <1>       ;cmc
7724                              <1>       ;jnc   loc_file_rw_restore_retn
7725 0000A06C 3C4E                <1>       cmp    al, 'N' ; 'no'
7726 0000A06E 0F840FF3FFFF        <1>       je     loc_file_rw_restore_retn
7727                              <1>
7728                              <1> copy_source_file_to_destination_pass_q:
7729 0000A074 B002                <1>       mov    al, 2  ; copy procedure Phase 2
7730 0000A076 E80C1C0000          <1>       call   copy_source_file_to_destination_file
7731                              <1>       ;jc    short loc_file_write_check_disk_space_err
7732                              <1>
7733                              <1>       ; 24/03/2016
7734                              <1>       ;push  cx
7735 0000A07B 51                  <1>       push   ecx ; 29/12/2017
7736 0000A07C BE[473F0100]        <1>       mov    esi, msg_copy_nextline
7737 0000A081 E8BFCFFFFF          <1>       call   print_msg
7738 0000A086 58                  <1>       pop    eax ; 29/12/2017
7739                              <1>       ;;pop  cx
7740                              <1>       ;pop   ax
7741                              <1>
7742                              <1>       ;or    cl, cl
7743 0000A087 08C0                <1>       or     al, al
7744 0000A089 7419                <1>       jz     short copy_source_file_to_destination_OK
7745                              <1>
7746                              <1>       ; 15/10/2016 (1Dh -> 18)
7747                              <1>       ; 18/03/2016 (1Dh)
7748                              <1>       ;cmp   cl, 18 ; write error
7749 0000A08B 3C12                <1>       cmp    al, 18
7750 0000A08D 7506                <1>       jne    short copy_source_file_to_destination_not_OK
7751                              <1>       ;
7752                              <1>       ;mov   al, cl ; error number (write fault!)
7753 0000A08F F9                  <1>       stc
7754 0000A090 E9EEF2FFFF          <1>       jmp    loc_file_rw_cmd_failed
7755                              <1>
7756                              <1> copy_source_file_to_destination_not_OK:
7757 0000A095 BE[823F0100]        <1>       mov    esi, Msg_read_file_error_before_EOF
7758 0000A09A E8A6CFFFFF          <1>       call   print_msg
7759 0000A09F E9DFF2FFFF          <1>       jmp    loc_file_rw_restore_retn
7760                              <1>
7761                              <1> copy_source_file_to_destination_OK:
7762 0000A0A4 BE[E73D0100]        <1>       mov    esi, Msg_OK
7763 0000A0A9 E897CFFFFF          <1>       call   print_msg
7764                              <1>
7765 0000A0AE E9D0F2FFFF          <1>       jmp    loc_file_rw_restore_retn
7766                              <1>
7767                              <1> ;loc_file_write_check_disk_space_err:
7768                              <1>       ;cmp   al, 27h ; Insufficient disk space
7769                              <1>       ;je    loc_file_write_insuff_disk_space_msg
7770                              <1>       ;jb    loc_file_rw_cmd_failed
7771                              <1>
7772                              <1>       ;call  print_misc_error_msg ; 15/03/2016
7773                              <1>       ;jmp   loc_file_rw_restore_retn
7774                              <1>
7775                              <1> change_fs_file_attributes:
7776                              <1>       ; 04/03/2016 ; Temporary
```

```
7777                                <1>       ; AL = File or directory attributes
7778                                <1>       ; AH = 0 -> Attributes are in MS-DOS format
7779                                <1>       ; AH > 0 -> Attributes are in SINGLIX format
7780                                <1>       ;push  ebx
7781                                <1>       ; ... do somethings here ...
7782                                <1>       ;pop   ebx
7783                                <1>       ; BL = File or directory attributes
7784 0000A0B3 C3                   <1>       retn
7785                                <1>
7786                                <1> set_get_env:
7787                                <1>       ; 11/04/2016 (TRDOS 386 = TRDOS v2.0)
7788                                <1>       ; 02/09/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_set')
7789                                <1>       ; 2005 - 28/08/2011
7790                                <1> get_setenv_fchar:
7791                                <1>       ; esi = environment variable/string
7792 0000A0B4 8A06                 <1>       mov    al, [esi]
7793 0000A0B6 3C20                 <1>       cmp    al, 20h
7794 0000A0B8 771E                 <1>       ja     short loc_find_env
7795                                <1>
7796 0000A0BA BE00300900           <1>       mov    esi, Env_Page
7797                                <1> loc_print_setline:
7798 0000A0BF 803E00               <1>       cmp    byte [esi], 0
7799 0000A0C2 7613                 <1>       jna    short loc_setenv_retn
7800 0000A0C4 E87CCFFFFF           <1>       call   print_msg
7801 0000A0C9 56                   <1>       push   esi
7802 0000A0CA BE[D7430100]         <1>       mov    esi, nextline
7803 0000A0CF E871CFFFFF           <1>       call   print_msg
7804 0000A0D4 5E                   <1>       pop    esi
7805 0000A0D5 EBE8                 <1>       jmp    short loc_print_setline
7806                                <1>
7807                                <1> loc_setenv_retn:
7808 0000A0D7 C3                   <1>       retn
7809                                <1>
7810                                <1> loc_find_env:
7811 0000A0D8 3C3D                 <1>       cmp    al, '='
7812 0000A0DA 0F8465E9FFFF         <1>       je     loc_cmd_failed
7813                                <1>
7814 0000A0E0 56                   <1>       push   esi
7815                                <1> loc_repeat_env_equal_check:
7816 0000A0E1 46                   <1>       inc    esi
7817 0000A0E2 803E3D               <1>       cmp    byte [esi], '='
7818 0000A0E5 7431                 <1>       je     short pass_env_equal_check
7819 0000A0E7 803E20               <1>       cmp    byte [esi], 20h
7820 0000A0EA 73F5                 <1>       jnb    short loc_repeat_env_equal_check
7821 0000A0EC C60600               <1>       mov    byte [esi], 0
7822 0000A0EF 5E                   <1>       pop    esi
7823 0000A0F0 BF[6E820100]         <1>       mov    edi, TextBuffer ; out buffer
7824 0000A0F5 B9FF000000           <1>       mov    ecx, 255 ; maximum size (limit)
7825 0000A0FA 30C0                 <1>       xor    al, al ; 0 -> use [ESI]
7826 0000A0FC E89E000000           <1>       call   get_environment_string
7827 0000A101 72D4                 <1>       jc     short loc_setenv_retn
7828                                <1>
7829 0000A103 BE[6E820100]         <1>       mov    esi, TextBuffer
7830 0000A108 E838CFFFFF           <1>       call   print_msg
7831 0000A10D BE[D7430100]         <1>       mov    esi, nextline
7832 0000A112 E82ECFFFFF           <1>       call   print_msg
7833                                <1>
7834 0000A117 C3                   <1>       retn
7835                                <1>
7836                                <1> pass_env_equal_check:
7837 0000A118 46                   <1>       inc    esi
7838 0000A119 803E20               <1>       cmp    byte [esi], 20h
7839 0000A11C 73FA                 <1>       jnb    short pass_env_equal_check
7840 0000A11E C60600               <1>       mov    byte [esi], 0
7841                                <1>
7842                                <1> loc_call_set_env_string:
7843 0000A121 5E                   <1>       pop    esi
7844 0000A122 E83B010000           <1>       call   set_environment_string
7845 0000A127 73AE                 <1>       jnc    short loc_setenv_retn
7846                                <1>
7847                                <1> loc_set_cmd_failed:
7848 0000A129 3C08                 <1>       cmp    al, 08h
7849 0000A12B 0F8514E9FFFF         <1>       jne    loc_cmd_failed
7850                                <1>
7851 0000A131 BE[C23F0100]         <1>       mov    esi, Msg_No_Set_Space
7852 0000A136 E80ACFFFFF           <1>       call   print_msg
7853                                <1>
7854 0000A13B C3                   <1>       retn
7855                                <1>
7856                                <1> set_get_path:
7857                                <1>       ; 11/04/2016 (TRDOS 386 = TRDOS v2.0)
7858                                <1>       ; 03/09/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_path')
7859                                <1>       ; 2005
7860                                <1> get_path_fchar:
7861                                <1>       ; esi = path
7862 0000A13C 803E20               <1>       cmp    byte [esi], 20h
7863 0000A13F 7737                 <1>       ja     short loc_set_path
7864                                <1>
7865 0000A141 BE00300900           <1>       mov    esi, Env_Page
7866                                <1> loc_print_path:
7867 0000A146 803E00               <1>       cmp    byte [esi], 0
7868 0000A149 762C                 <1>       jna    short loc_path_retn
7869                                <1>
7870 0000A14B BE[213A0100]         <1>       mov    esi, Cmd_Path ; 'PATH' address
7871 0000A150 BF[6E820100]         <1>       mov    edi, TextBuffer ; out buffer
7872 0000A155 30C0                 <1>       xor    al, al  ; use [ESI]
7873 0000A157 B9FF000000           <1>       mov    ecx, 255 ; maximum size (limit)
7874 0000A15C E83E000000           <1>       call   get_environment_string
7875 0000A161 7214                 <1>       jc     short loc_path_retn
7876                                <1>
7877 0000A163 BE[6E820100]         <1>       mov    esi, TextBuffer
7878 0000A168 E8D8CEFFFF           <1>       call   print_msg
7879 0000A16D BE[D7430100]         <1>       mov    esi, nextline
7880 0000A172 E8CECEFFFF           <1>       call   print_msg
7881                                <1>
```

```
                                <1> loc_path_retn:
0000A177 C3                     <1>         retn
                                <1>
                                <1> loc_set_path:
0000A178 56                     <1>         push   esi
                                <1> loc_set_path_find_end:
0000A179 46                     <1>         inc    esi
0000A17A 803E20                 <1>         cmp    byte [esi], 20h
0000A17D 73FA                   <1>         jnb    short loc_set_path_find_end
0000A17F C60600                 <1>         mov    byte [esi], 0
                                <1> loc_set_path_header:
0000A182 5E                     <1>         pop    esi
                                <1> set_path_x: ; 31/12/2017 ('syspath')
0000A183 4E                     <1>         dec    esi
0000A184 C6063D                 <1>         mov    byte [esi], '='
0000A187 4E                     <1>         dec    esi
0000A188 C60648                 <1>         mov    byte [esi], 'H'
0000A18B 4E                     <1>         dec    esi
0000A18C C60654                 <1>         mov    byte [esi], 'T'
0000A18F 4E                     <1>         dec    esi
0000A190 C60641                 <1>         mov    byte [esi], 'A'
0000A193 4E                     <1>         dec    esi
0000A194 C60650                 <1>         mov    byte [esi], 'P'
                                <1>
                                <1> loc_path_call_set_env_string:
0000A197 E8C6000000             <1>         call   set_environment_string
0000A19C 728B                   <1>         jc     short loc_set_cmd_failed
                                <1>
0000A19E C3                     <1>         retn
                                <1>
                                <1> get_environment_string:
                                <1>         ; 12/04/2016
                                <1>         ; 11/04/2016
                                <1>         ; 05/04/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>         ; 02/09/2011 (TRDOS v1, MAINPROG.ASM)
                                <1>         ; 28/08/2011
                                <1>         ; INPUT->
                                <1>         ;      EDI = Output buffer
                                <1>         ;      CX = Buffer length (<= ENV_PAGE_SIZE)
                                <1>         ;
                                <1>         ;      AL > 0 = AL = String sequence number
                                <1>         ;      AL = 0 -> ESI = ASCIIZ Set word
                                <1>         ;             (environment variable)
                                <1>         ; OUTPUT ->
                                <1>         ;      ESI is not changed
                                <1>         ;      EDI is not changed
                                <1>         ;      EAX = String length (with zero tail)
                                <1>         ;      EDX = Environment variables page address
                                <1>         ;      CF = 1 -> Not found (EAX not valid)
                                <1>         ;
                                <1>         ; (Modified registers: EAX, EDX)
                                <1>
0000A19F BA00300900             <1>         mov    edx, Env_Page
0000A1A4 803A00                 <1>         cmp    byte [edx], 0
0000A1A7 7474                   <1>         jz     short get_env_string_with_word_stc_retn
                                <1>
0000A1A9 66890D[D88C0100]       <1>         mov    [env_var_length], cx
                                <1>
0000A1B0 51                     <1>         push   ecx ; *
0000A1B1 56                     <1>         push   esi ; **
                                <1>
0000A1B2 08C0                   <1>         or     al, al
0000A1B4 7449                   <1>         jz     short get_env_string_with_word
                                <1>
                                <1> get_env_string_with_seq_number:
0000A1B6 B101                   <1>         mov    cl, 1
0000A1B8 88C5                   <1>         mov    ch, al
0000A1BA 31C0                   <1>         xor    eax, eax
0000A1BC 89D6                   <1>         mov    esi, edx ; Env_Page
                                <1>
                                <1> get_env_string_seq_number_check:
0000A1BE 38CD                   <1>         cmp    ch, cl
0000A1C0 7726                   <1>         ja     short get_env_string_seq_number_next
                                <1>
                                <1> get_env_string_move_to_buff:
0000A1C2 57                     <1>         push   edi ; ***
                                <1>
0000A1C3 29D2                   <1>         sub    edx, edx
                                <1>
                                <1> get_env_string_seq_number_repeat1:
0000A1C5 42                     <1>         inc    edx
0000A1C6 AC                     <1>         lodsb
0000A1C7 AA                     <1>         stosb
                                <1>
0000A1C8 66FF0D[D88C0100]       <1>         dec    word [env_var_length]
0000A1CF 7508                   <1>         jnz    short get_env_string_seq_number_repeat3
                                <1>
                                <1> get_env_string_seq_number_repeat2:
0000A1D1 20C0                   <1>         and    al, al
0000A1D3 7408                   <1>         jz     short get_env_string_seq_number_ok
0000A1D5 42                     <1>         inc    edx
0000A1D6 AC                     <1>         lodsb
0000A1D7 EBF8                   <1>         jmp    short get_env_string_seq_number_repeat2
                                <1>
                                <1> get_env_string_seq_number_repeat3:
0000A1D9 08C0                   <1>         or     al, al
0000A1DB 75E8                   <1>         jnz    short get_env_string_seq_number_repeat1
                                <1>
                                <1> get_env_string_seq_number_ok:
0000A1DD 5F                     <1>         pop    edi ; ***
0000A1DE 89D0                   <1>         mov    eax, edx ; Length of the environment string
                                <1>                         ; (ASCIIZ, includes ZERO tail)
0000A1E0 BA00300900             <1>         mov    edx, Env_Page
                                <1>
                                <1> get_env_string_stc_retn:
```

```
7987 0000A1E5 5E                <1>       pop    esi ; **
7988 0000A1E6 59                <1>       pop    ecx ; *
7989 0000A1E7 C3                <1>       retn
7990                            <1>
7991                            <1> get_env_string_seq_number_next:
7992 0000A1E8 AC                <1>       lodsb
7993 0000A1E9 08C0              <1>       or     al, al
7994 0000A1EB 75FB              <1>       jnz    short get_env_string_seq_number_next
7995                            <1>
7996 0000A1ED 81FE00320900      <1>       cmp    esi, Env_Page + Env_Page_Size ; +512 (+4096)
7997 0000A1F3 F5                <1>       cmc
7998 0000A1F4 72EF              <1>       jc     short get_env_string_stc_retn
7999                            <1>
8000 0000A1F6 AC                <1>       lodsb
8001 0000A1F7 3C01              <1>       cmp    al, 1
8002 0000A1F9 72EA              <1>       jb     short get_env_string_stc_retn
8003 0000A1FB FEC1              <1>       inc    cl
8004 0000A1FD EBBF              <1>       jmp    short get_env_string_seq_number_check
8005                            <1>
8006                            <1> get_env_string_with_word:
8007 0000A1FF 31C9              <1>       xor    ecx, ecx
8008                            <1>
8009                            <1> get_env_string_calc_word_length:
8010 0000A201 AC                <1>       lodsb
8011 0000A202 3C20              <1>       cmp    al, 20h
8012 0000A204 7211              <1>       jb     short get_env_string_calc_word_length_ok
8013                            <1>       ;inc   cx
8014 0000A206 FEC1              <1>       inc    cl
8015                            <1>
8016 0000A208 3C61              <1>       cmp    al, 'a'
8017 0000A20A 72F5              <1>       jb     short get_env_string_calc_word_length
8018 0000A20C 3C7A              <1>       cmp    al, 'z'
8019 0000A20E 77F1              <1>       ja     short get_env_string_calc_word_length
8020 0000A210 24DF              <1>       and    al, 0DFh
8021 0000A212 8846FF            <1>       mov    [esi-1], al
8022 0000A215 EBEA              <1>       jmp    short get_env_string_calc_word_length
8023                            <1>
8024                            <1> get_env_string_calc_word_length_ok:
8025 0000A217 08C9              <1>       or     cl, cl
8026 0000A219 7506              <1>       jnz    short get_env_string_calc_word_length_save
8027                            <1>
8028 0000A21B 5E                <1>       pop    esi ; **
8029                            <1>
8030                            <1> get_env_string_stc_retn1:
8031 0000A21C 59                <1>       pop    ecx ; *
8032                            <1>
8033                            <1> get_env_string_with_word_stc_retn:
8034 0000A21D 31C0              <1>       xor    eax, eax
8035 0000A21F F9                <1>       stc
8036 0000A220 C3                <1>       retn
8037                            <1>
8038                            <1> get_env_string_calc_word_length_save:
8039 0000A221 871C24            <1>       xchg   ebx, [esp] ; **
8040 0000A224 89DE              <1>       mov    esi, ebx
8041                            <1>              ; Start of the env string (to be searched)
8042                            <1>
8043 0000A226 57                <1>       push   edi ; ***
8044 0000A227 89D7              <1>       mov    edi, edx ; Env_Page
8045                            <1>
8046                            <1> get_env_string_compare:
8047 0000A229 57                <1>       push   edi ; ****
8048 0000A22A 51                <1>       push   ecx ; ***** ; Variable name length
8049                            <1>
8050                            <1> get_env_string_compare_rep:
8051 0000A22B AC                <1>       lodsb
8052 0000A22C AE                <1>       scasb
8053 0000A22D 7511              <1>       jne    short get_env_string_compare_next1
8054 0000A22F E2FA              <1>       loop   get_env_string_compare_rep
8055                            <1>
8056 0000A231 803F3D            <1>       cmp    byte [edi], '='
8057 0000A234 750A              <1>       jne    short get_env_string_compare_next1
8058                            <1>
8059 0000A236 59                <1>       pop    ecx ; *****
8060 0000A237 5F                <1>       pop    edi ; ****
8061 0000A238 89FE              <1>       mov    esi, edi
8062 0000A23A 5F                <1>       pop    edi ; ***
8063 0000A23B 871C24            <1>       xchg   ebx, [esp] ; **
8064 0000A23E EB82              <1>       jmp    short get_env_string_move_to_buff
8065                            <1>
8066                            <1> get_env_string_compare_next1:
8067 0000A240 89FE              <1>       mov    esi, edi
8068 0000A242 59                <1>       pop    ecx ; *****
8069 0000A243 5F                <1>       pop    edi ; ****
8070                            <1> get_env_string_compare_next2:
8071 0000A244 81FEFF310900      <1>       cmp    esi, Env_Page + Env_Page_Size - 1 ; +511 (+4095)
8072 0000A24A 7310              <1>       jnb    short get_env_string_compare_not_ok
8073 0000A24C 20C0              <1>       and    al, al
8074 0000A24E AC                <1>       lodsb
8075 0000A24F 75F3              <1>       jnz    short get_env_string_compare_next2
8076 0000A251 08C0              <1>       or     al, al
8077 0000A253 7407              <1>       jz     short get_env_string_compare_not_ok
8078 0000A255 4E                <1>       dec    esi ; 12/04/2016
8079 0000A256 89F7              <1>       mov    edi, esi
8080 0000A258 89DE              <1>       mov    esi, ebx
8081 0000A25A EBCD              <1>       jmp    short get_env_string_compare
8082                            <1>
8083                            <1> get_env_string_compare_not_ok:
8084 0000A25C 5F                <1>       pop    edi ; ***
8085 0000A25D 89DE              <1>       mov    esi, ebx
8086 0000A25F 5B                <1>       pop    ebx ; **
8087 0000A260 EBBA              <1>       jmp    short get_env_string_stc_retn1
8088                            <1>
8089                            <1> set_environment_string:
8090                            <1>       ; 13/04/2016
8091                            <1>       ; 12/04/2016
```

```
8092                            <1>       ; 11/04/2016
8093                            <1>       ; 06/04/2016
8094                            <1>       ; 05/04/2016 (TRDOS 386 = TRDOS v2.0)
8095                            <1>       ; 02/09/2011 (TRDOS v1, MAINPROG.ASM)
8096                            <1>       ; 29/08/2011
8097                            <1>       ; 29/08/2011
8098                            <1>       ; INPUT->
8099                            <1>       ;      ESI = ASCIIZ environment string
8100                            <1>       ; OUTPUT ->
8101                            <1>       ;      ESI is not changed
8102                            <1>       ;      CF = 1 -> Could not set,
8103                            <1>       ;           insufficient environment space
8104                            <1>       ;
8105                            <1>       ; (EAX, EDX will be changed)
8106                            <1>       ;
8107                            <1>       ;    (EAX = Start address of the env string if > 0)
8108                            <1>       ;    (EDX = Environment string length)
8109                            <1>
8110 0000A262 56               <1>       push   esi ; *
8111                            <1>
8112 0000A263 31C0             <1>       xor    eax, eax
8113                            <1>
8114                            <1> set_env_chk_validation1:
8115 0000A265 FEC4             <1>       inc    ah ; variable (string) length
8116 0000A267 AC               <1>       lodsb
8117 0000A268 3C3D             <1>       cmp    al, '='
8118 0000A26A 7415             <1>       je     short set_env_chk_validation2
8119 0000A26C 3C20             <1>       cmp    al, 20h
8120 0000A26E 720F             <1>       jb     short set_env_string_stc
8121                            <1>
8122                            <1>       ; 06/04/2016
8123 0000A270 3C61             <1>       cmp    al, 'a'
8124 0000A272 72F1             <1>       jb     short set_env_chk_validation1
8125 0000A274 3C7A             <1>       cmp    al, 'z'
8126 0000A276 77ED             <1>       ja     short set_env_chk_validation1
8127 0000A278 2C20             <1>       sub    al, 'a'-'A'
8128 0000A27A 8846FF           <1>       mov    [esi-1], al
8129 0000A27D EBE6             <1>       jmp    short set_env_chk_validation1
8130                            <1>
8131                            <1> set_env_string_stc:
8132 0000A27F 5E               <1>       pop    esi ; *
8133                            <1>       ;stc
8134 0000A280 C3               <1>       retn
8135                            <1>
8136                            <1> set_env_chk_validation2:
8137 0000A281 51               <1>       push   ecx ; **
8138 0000A282 53               <1>       push   ebx ; ***
8139 0000A283 57               <1>       push   edi ; ****
8140                            <1>
8141                            <1>       ; 12/04/2016
8142 0000A284 8B5C240C         <1>       mov    ebx, [esp+12]
8143                            <1>
8144                            <1> set_env_chk_validation2w:
8145 0000A288 89F7             <1>       mov    edi, esi
8146 0000A28A 4F               <1>       dec    edi
8147                            <1>
8148 0000A28B 807FFF20         <1>       cmp    byte [edi-1], 20h
8149 0000A28F 771A             <1>       ja     short set_env_chk_validation2z
8150                            <1>
8151 0000A291 56               <1>       push   esi
8152 0000A292 89FE             <1>       mov    esi, edi
8153 0000A294 4E               <1>       dec    esi
8154                            <1>
8155                            <1> set_env_chk_validation2x:
8156 0000A295 4E               <1>       dec    esi
8157                            <1>
8158 0000A296 39DE             <1>       cmp    esi, ebx
8159 0000A298 7207             <1>       jb     short set_env_chk_validation2y
8160                            <1>
8161 0000A29A 4F               <1>       dec    edi
8162                            <1>
8163 0000A29B 8A06             <1>       mov    al, [esi]
8164 0000A29D 8807             <1>       mov    [edi], al
8165                            <1>
8166 0000A29F EBF4             <1>       jmp    short set_env_chk_validation2x
8167                            <1>
8168                            <1> set_env_chk_validation2y:
8169 0000A2A1 5E               <1>       pop    esi
8170                            <1>
8171                            <1>       ;mov   byte [ebx], 20h
8172                            <1>
8173 0000A2A2 43               <1>       inc    ebx
8174 0000A2A3 895C240C         <1>       mov    [esp+12], ebx
8175                            <1>
8176 0000A2A7 FECC             <1>       dec    ah ; 13/04/2016
8177                            <1>
8178 0000A2A9 EBDD             <1>       jmp    short set_env_chk_validation2w
8179                            <1>
8180                            <1> set_env_chk_validation2z:
8181 0000A2AB BA00300900       <1>       mov    edx, Env_Page
8182 0000A2B0 89D7             <1>       mov    edi, edx
8183                            <1>
8184                            <1> set_env_chk_validation3:
8185 0000A2B2 AC               <1>       lodsb
8186 0000A2B3 3C20             <1>       cmp    al, 20h
8187 0000A2B5 74FB             <1>       je     short set_env_chk_validation3
8188                            <1>
8189 0000A2B7 9C               <1>       pushf
8190                            <1>
8191                            <1>       ; 12/04/2016
8192                            <1> set_env_chk_validation3n:
8193 0000A2B8 3C61             <1>       cmp    al, 'a'
8194 0000A2BA 720C             <1>       jb     short set_env_chk_validation3c
8195 0000A2BC 3C7A             <1>       cmp    al, 'z'
8196 0000A2BE 7705             <1>       ja     short set_env_chk_validation3x
```

```
8197 0000A2C0 2C20              <1>       sub    al, 'a'-'A'
8198 0000A2C2 8846FF            <1>       mov    [esi-1], al
8199                            <1>
8200                            <1> set_env_chk_validation3x:
8201 0000A2C5 AC                <1>       lodsb
8202 0000A2C6 EBF0              <1>       jmp    short set_env_chk_validation3n
8203                            <1>
8204                            <1> set_env_chk_validation3c:
8205 0000A2C8 3C20              <1>       cmp    al, 20h
8206 0000A2CA 73F9              <1>       jnb    short set_env_chk_validation3x
8207                            <1>
8208 0000A2CC 803F00            <1>       cmp    byte [edi], 0
8209 0000A2CF 7731              <1>       ja     short set_env_chk_validation4
8210                            <1>
8211 0000A2D1 9D                <1>       popf
8212 0000A2D2 7228              <1>       jb     short set_env_string_nothing
8213                            <1>
8214 0000A2D4 B900020000        <1>       mov    ecx, Env_Page_Size ; 512 (4096)
8215                            <1>
8216 0000A2D9 89DE              <1>       mov    esi, ebx ; 12/04/2016
8217                            <1>
8218                            <1> set_env_string_copy_to_envb:
8219 0000A2DB AC                <1>       lodsb
8220 0000A2DC 3C20              <1>       cmp    al, 20h
8221 0000A2DE 720A              <1>       jb     short set_env_string_copy_to_envb_z
8222 0000A2E0 AA                <1>       stosb
8223 0000A2E1 E2F8              <1>       loop   set_env_string_copy_to_envb
8224                            <1>
8225                            <1>       ; 11/04/2016
8226 0000A2E3 89D7              <1>       mov    edi, edx ; Env_Page
8227 0000A2E5 B900020000        <1>       mov    ecx, Env_Page_Size
8228                            <1>
8229                            <1> set_env_string_copy_to_envb_z:
8230 0000A2EA 52                <1>       push   edx  ; Start address of the variable
8231 0000A2EB BA00020000        <1>       mov    edx, Env_Page_Size
8232 0000A2F0 29CA              <1>       sub    edx, ecx ; variable (string) length
8233                            <1>
8234 0000A2F2 28C0              <1>       sub    al, al ; 0
8235 0000A2F4 F3AA              <1>       rep    stosb ; clear remain bytes of the env page
8236                            <1>
8237 0000A2F6 58                <1>       pop    eax  ; Start address of the variable
8238                            <1>
8239                            <1> set_env_string_allocate_envb_retn:  ; stc or clc return
8240 0000A2F7 5F                <1>       pop    edi ; ****
8241 0000A2F8 5B                <1>       pop    ebx ; ***
8242 0000A2F9 59                <1>       pop    ecx ; **
8243 0000A2FA 5E                <1>       pop    esi ; *
8244 0000A2FB C3                <1>       retn
8245                            <1>
8246                            <1> set_env_string_nothing:
8247 0000A2FC 31C0              <1>       xor    eax, eax
8248 0000A2FE 31D2              <1>       xor    edx, edx ; 11/04/2016
8249 0000A300 EBF5              <1>       jmp    short set_env_string_allocate_envb_retn
8250                            <1>
8251                            <1> set_env_chk_validation4:
8252                            <1>       ; 11/04/2016
8253 0000A302 9D                <1>       popf
8254                            <1>
8255 0000A303 89D6              <1>       mov    esi, edx  ; Env_Page
8256                            <1>
8257                            <1> set_env_chk_validation5:
8258 0000A305 89DF              <1>       mov    edi, ebx  ; ASCIIZ environment string address
8259 0000A307 0FB6CC            <1>       movzx  ecx, ah ; Variable (string) length (with '=')
8260                            <1>
8261                            <1> set_env_chk_validation5_loop:
8262 0000A30A AC                <1>       lodsb
8263 0000A30B AE                <1>       scasb
8264 0000A30C 750A              <1>       jne    short set_env_chk_validation6
8265 0000A30E E2FA              <1>       loop   set_env_chk_validation5_loop
8266                            <1>
8267 0000A310 3C3D              <1>       cmp    al, '='
8268 0000A312 0F8483000000      <1>       je    set_env_change_variable
8269                            <1>
8270                            <1> set_env_chk_validation6:
8271 0000A318 08C0              <1>       or     al, al ; 0
8272 0000A31A 7403              <1>       jz     short set_env_chk_validation7
8273                            <1>
8274 0000A31C AC                <1>       lodsb
8275 0000A31D EBF9              <1>       jmp    short set_env_chk_validation6
8276                            <1>
8277                            <1> set_env_chk_validation7:
8278 0000A31F 88E1              <1>       mov    cl, ah
8279 0000A321 01F1              <1>       add    ecx, esi
8280 0000A323 81F9FF310900      <1>       cmp    ecx, Env_Page + Env_Page_Size - 1
8281                            <1>              ; 511 (4095)
8282                            <1>              ; strlen + '=' + 0
8283 0000A329 72DA              <1>       jb     short set_env_chk_validation5
8284                            <1>
8285                            <1> set_env_chk_validation8: ; variable not found
8286 0000A32B 0FB6F4            <1>       movzx  esi, ah  ; variable name length (with '=')
8287 0000A32E 01DE              <1>       add    esi, ebx ; position just after of the '='
8288                            <1>
8289                            <1> set_env_chk_validation8_loop:
8290 0000A330 AC                <1>       lodsb
8291 0000A331 3C20              <1>       cmp    al, 20h
8292 0000A333 74FB              <1>       je     short set_env_chk_validation8_loop
8293 0000A335 72C5              <1>       jb     short set_env_string_nothing
8294                            <1>
8295                            <1> set_env_chk_validation9:
8296 0000A337 AC                <1>       lodsb
8297 0000A338 3C20              <1>       cmp    al, 20h
8298 0000A33A 73FB              <1>       jnb    short set_env_chk_validation9
8299                            <1>
8300                            <1>       ; End of ASCIIZ environment string
8301                            <1>
```

```
8302                            <1> set_env_add_variable:
8303 0000A33C 29DE             <1>       sub    esi, ebx ; variable+definition length
8304                            <1>
8305 0000A33E 56               <1>       push   esi ; *****
8306                            <1>
8307 0000A33F 89D6             <1>       mov    esi, edx ; Environment page address
8308                            <1>
8309 0000A341 B900020000       <1>       mov    ecx, Env_Page_Size ; 512 (4096)
8310                            <1>
8311                            <1> set_env_add_variable_loop:
8312 0000A346 AC               <1>       lodsb
8313 0000A347 20C0             <1>       and    al, al
8314 0000A349 7406             <1>       jz     short set_env_add_variable_chk1 ; 0
8315 0000A34B E2F9             <1>       loop   set_env_add_variable_loop
8316                            <1>
8317                            <1>       ; 11/04/2016
8318 0000A34D 884EFF           <1>       mov    [esi-1], cl ; 0
8319 0000A350 41               <1>       inc    ecx
8320                            <1>
8321                            <1> set_env_add_variable_chk1:
8322 0000A351 49               <1>       dec    ecx
8323 0000A352 7408             <1>       jz     short set_env_add_variable_nspc
8324 0000A354 AC               <1>       lodsb
8325 0000A355 08C0             <1>       or     al, al
8326 0000A357 740C             <1>       jz     short set_env_add_variable_chk2 ; 00
8327 0000A359 49               <1>       dec    ecx
8328 0000A35A 75EA             <1>       jnz    short set_env_add_variable_loop
8329                            <1>
8330                            <1> set_env_add_variable_nspc: ; no space on environment page
8331 0000A35C 58               <1>       pop    eax ; *****
8332 0000A35D B808000000       <1>       mov    eax, 8 ; No space for new environment string
8333 0000A362 F9               <1>       stc
8334 0000A363 EB92             <1>         jmp     short set_env_string_allocate_envb_retn
8335                            <1>
8336                            <1> set_env_add_variable_chk2:
8337 0000A365 8B0C24           <1>       mov    ecx, [esp] ; *****
8338 0000A368 4E               <1>       dec    esi ; beginning address of the new variable
8339 0000A369 89F0             <1>       mov    eax, esi
8340 0000A36B 01C8             <1>       add    eax, ecx ; string length (with CR)
8341 0000A36D 81C200020000     <1>       add    edx, Env_Page_Size ; 512 (4096)
8342 0000A373 39D0             <1>       cmp    eax, edx
8343 0000A375 77E5             <1>       ja     short set_env_add_variable_nspc
8344 0000A377 49               <1>       dec    ecx ; except CR at the end
8345 0000A378 89CA             <1>       mov    edx, ecx ; 12/04/2016
8346 0000A37A 89F7             <1>       mov    edi, esi
8347 0000A37C 893C24           <1>       mov    [esp], edi ; ***** ; Start address of new variable
8348 0000A37F 89DE             <1>       mov    esi, ebx ; ASCIIZ environment string address
8349 0000A381 F3A4             <1>       rep    movsb
8350 0000A383 28C0             <1>       sub    al, al
8351 0000A385 AA               <1>       stosb
8352 0000A386 58               <1>       pop    eax ; ***** ; Beginning address of new variable
8353 0000A387 81FF00320900     <1>       cmp    edi, Env_Page + Env_Page_Size ; 12/04/2016
8354 0000A38D 0F8364FFFFFF     <1>       jnb    set_env_string_allocate_envb_retn ; OK !
8355 0000A393 880F             <1>       mov    [edi], cl ; 0
8356 0000A395 F8               <1>       clc    ; 13/04/2016
8357 0000A396 E95CFFFFFF       <1>       jmp    set_env_string_allocate_envb_retn ; OK !
8358                            <1>
8359                            <1> set_env_change_variable:
8360                            <1>       ; 06/04/2016
8361                            <1>       ; esi = Variable's address in environment page (after '=')
8362                            <1>       ; edi = ASCIIZ environment string address (after '=')
8363                            <1>
8364                            <1>       ; ah = variable length from start to the '='
8365 0000A39B 8825[D88C0100]     <1>       mov    [env_var_length], ah
8366                            <1>
8367 0000A3A1 28C9             <1>       sub    cl, cl ; ecx = 0
8368                            <1>
8369 0000A3A3 57               <1>       push   edi ; *****
8370                            <1>
8371 0000A3A4 89F7             <1>       mov    edi, esi ; 11/04/2016
8372                            <1>
8373                            <1> set_env_change_variable_calc1:
8374 0000A3A6 AC               <1>       lodsb
8375 0000A3A7 08C0             <1>       or     al, al
8376 0000A3A9 7403             <1>       jz     short set_env_change_variable_calc2
8377                            <1>
8378 0000A3AB 41               <1>       inc    ecx ; length of environment string (after the '=')
8379                            <1>
8380 0000A3AC EBF8             <1>       jmp    short set_env_change_variable_calc1
8381                            <1>
8382                            <1> set_env_change_variable_calc2:
8383 0000A3AE 8B3424           <1>       mov    esi, [esp] ; ASCIIZ environment string address
8384                            <1>
8385 0000A3B1 29D2             <1>       sub    edx, edx
8386                            <1>
8387                            <1> set_env_change_variable_calc3:
8388 0000A3B3 AC               <1>       lodsb
8389 0000A3B4 3C20             <1>       cmp    al, 20h
8390 0000A3B6 7203             <1>       jb     short set_env_change_variable_calc4
8391                            <1>
8392 0000A3B8 42               <1>       inc    edx ; length of ASCIIZ string (after the '=')
8393                            <1>
8394 0000A3B9 EBF8             <1>       jmp    short set_env_change_variable_calc3
8395                            <1>
8396                            <1> set_env_change_variable_calc4:
8397 0000A3BB C646FF00         <1>       mov    byte [esi-1], 0  ; put ZERO instead of CR
8398                            <1>
8399 0000A3BF 5E               <1>       pop    esi ; ***** ; ASCIIZ string address (after '=')
8400                            <1>
8401                            <1>       ; EDI = Old variable's address (after '=')
8402                            <1>
8403                            <1>       ; compare the new string with the old string
8404 0000A3C0 39CA             <1>       cmp    edx, ecx
8405 0000A3C2 7717             <1>       ja     short set_env_change_variable_calc5 ; longer
8406 0000A3C4 0F828F000000     <1>       jb    set_env_change_variable_calc9 ; shorter
```

```
8407                                <1>
8408                                <1>       ;same length (simple copy)
8409 0000A3CA 0FB6C4               <1>       movzx  eax, ah
8410 0000A3CD 01C2                 <1>       add    edx, eax
8411 0000A3CF F7D8                 <1>       neg    eax
8412 0000A3D1 01F8                 <1>       add    eax, edi
8413                                <1>       ; EAX = Start address of the variable
8414                                <1>       ; EDX = Variable length (without ZERO at the end of variable)
8415                                <1>
8416 0000A3D3 F3A4                 <1>       rep    movsb
8417 0000A3D5 F8                   <1>       clc    ; 13/04/2016
8418 0000A3D6 E91CFFFFFF           <1>       jmp   set_env_string_allocate_envb_retn ; OK !
8419                                <1>
8420                                <1> set_env_change_variable_calc5:
8421                                <1>       ; 11/04/2016
8422 0000A3DB 52                   <1>       push   edx ; *****
8423 0000A3DC 29CA                 <1>       sub    edx, ecx ; difference ; (the new string is longer)
8424 0000A3DE 89F3                 <1>       mov    ebx, esi
8425 0000A3E0 89FE                 <1>       mov    esi, edi
8426                                <1>
8427                                <1> set_env_change_variable_calc6:
8428 0000A3E2 AC                   <1>       lodsb
8429 0000A3E3 20C0                 <1>       and    al, al
8430 0000A3E5 75FB                 <1>       jnz    short set_env_change_variable_calc6
8431                                <1>
8432 0000A3E7 81FE00320900         <1>       cmp    esi, Env_Page + Env_Page_Size ; 512 (4096)
8433 0000A3ED 0F8369FFFFFF         <1>       jnb   set_env_add_variable_nspc
8434                                <1>
8435 0000A3F3 89F9                 <1>       mov    ecx, edi  ; current (old) variable's address
8436 0000A3F5 89F7                 <1>       mov    edi, esi  ; next variable's address
8437                                <1>
8438 0000A3F7 AC                   <1>       lodsb
8439 0000A3F8 08C0                 <1>       or     al, al
8440 0000A3FA 7416                 <1>       jz     short set_env_change_variable_calc8 ; 00
8441                                <1>
8442                                <1> set_env_change_variable_calc7:
8443 0000A3FC AC                   <1>       lodsb
8444 0000A3FD 20C0                 <1>       and    al, al
8445 0000A3FF 75FB                 <1>       jnz    short set_env_change_variable_calc7
8446                                <1>
8447 0000A401 81FE00320900         <1>       cmp    esi, Env_Page + Env_Page_Size ; 512 (4096)
8448 0000A407 0F834FFFFFFF         <1>       jnb   set_env_add_variable_nspc
8449                                <1>
8450 0000A40D AC                   <1>       lodsb
8451 0000A40E 08C0                 <1>       or     al, al
8452 0000A410 75EA                 <1>       jnz    short set_env_change_variable_calc7
8453                                <1>
8454                                <1> set_env_change_variable_calc8:
8455 0000A412 4E                   <1>       dec    esi ; address of the second (last) 0 of the 00
8456                                <1>
8457 0000A413 01F2                 <1>       add    edx, esi ; final position of the last 0
8458                                <1>
8459 0000A415 81FA00320900         <1>       cmp    edx, Env_Page + Env_Page_Size ; 512 (4096)
8460 0000A41B 0F833BFFFFFF         <1>       jnb   set_env_add_variable_nspc
8461                                <1>
8462 0000A421 89C8                 <1>       mov    eax, ecx ; old variable's address (after '=')
8463                                <1>
8464 0000A423 89F1                 <1>       mov    ecx, esi
8465 0000A425 29F9                 <1>       sub    ecx, edi ; count of bytes to move forward
8466                                <1>
8467                                <1>       ; 13/04/2016
8468 0000A427 C60200               <1>       mov    byte [edx], 0
8469 0000A42A 89D7                 <1>       mov    edi, edx
8470 0000A42C 29F2                 <1>       sub    edx, esi ; difference (additional byte count)
8471 0000A42E 4F                   <1>       dec    edi ; the last zero address (first byte of the 00)
8472 0000A42F 89FE                 <1>       mov    esi, edi
8473 0000A431 29D6                 <1>       sub    esi, edx ; - displacement
8474                                <1>
8475 0000A433 FA                   <1>       cli    ; disable interrupts
8476 0000A434 FD                   <1>       std    ; backward
8477                                <1>
8478 0000A435 F3A4                 <1>       rep    movsb ; move ECX bytes from DS:ESI to ES:EDI
8479                                <1>
8480 0000A437 FC                   <1>       cld    ; forward (default)
8481 0000A438 FB                   <1>       sti    ; enable interrupts
8482                                <1>
8483 0000A439 89C7                 <1>       mov    edi, eax
8484 0000A43B 59                   <1>       pop    ecx ; ***** ; byte count (after '=')
8485 0000A43C 89CA                 <1>       mov    edx, ecx
8486 0000A43E 89DE                 <1>       mov    esi, ebx ; ASCIIZ string address (after '=')
8487 0000A440 89FB                 <1>       mov    ebx, edi
8488                                <1>
8489 0000A442 F3A4                 <1>       rep    movsb
8490                                <1>
8491 0000A444 880F                 <1>       mov    [edi], cl ; 0 ; end of variable
8492                                <1>
8493 0000A446 0FB605[D88C0100]     <1>       movzx  eax, byte [env_var_length]
8494 0000A44D 01C2                 <1>       add    edx, eax ; variable length (total)
8495 0000A44F F7D8                 <1>       neg    eax
8496 0000A451 01D8                 <1>       add    eax, ebx ; start address of the variable
8497 0000A453 F8                   <1>       clc    ; 13/04/2016
8498 0000A454 E99EFEFFFF           <1>       jmp   set_env_string_allocate_envb_retn ; OK !
8499                                <1>
8500                                <1> set_env_change_variable_calc9:
8501                                <1>       ; 11/04/2016
8502 0000A459 21D2                 <1>       and    edx, edx ; is empty ?
8503 0000A45B 753B                 <1>       jnz    short set_env_change_variable_calc15
8504                                <1>
8505 0000A45D 0FB6DC               <1>       movzx  ebx, ah
8506 0000A460 F7DB                 <1>       neg    ebx
8507 0000A462 01FB                 <1>       add    ebx, edi
8508                                <1>
8509                                <1>       ; EBX = Start address of the variable (in env page)
8510                                <1>       ; EDX = Variable length = 0
8511                                <1>
```

```
8512 0000A464 89FE              <1>       mov    esi, edi
8513                            <1>
8514                            <1> set_env_change_variable_calc10:
8515 0000A466 AC                <1>       lodsb
8516 0000A467 08C0              <1>       or     al, al
8517 0000A469 75FB              <1>       jnz    short set_env_change_variable_calc10
8518                            <1>
8519 0000A46B B9FF310900        <1>       mov    ecx, Env_Page + Env_Page_Size - 1
8520                            <1>
8521 0000A470 39CE              <1>       cmp    esi, ecx ; +511 (+4095)
8522 0000A472 7604              <1>       jna    short set_env_change_variable_calc11
8523                            <1>
8524 0000A474 89CE              <1>       mov    esi, ecx
8525 0000A476 8806              <1>       mov    [esi], al ; 0
8526                            <1>
8527                            <1> set_env_change_variable_calc11:
8528 0000A478 89DF              <1>       mov    edi, ebx ; old variable's start address
8529                            <1>
8530                            <1> set_env_change_variable_calc12:
8531 0000A47A AC                <1>       lodsb
8532 0000A47B AA                <1>       stosb
8533 0000A47C 20C0              <1>       and    al, al
8534 0000A47E 75FA              <1>       jnz    short set_env_change_variable_calc12
8535 0000A480 39CE              <1>       cmp    esi, ecx
8536 0000A482 7706              <1>       ja     short set_env_change_variable_calc13
8537 0000A484 AC                <1>       lodsb
8538 0000A485 AA                <1>       stosb
8539 0000A486 20C0              <1>       and    al, al
8540 0000A488 75F0              <1>       jnz    short set_env_change_variable_calc12
8541                            <1>
8542                            <1> set_env_change_variable_calc13:
8543 0000A48A 29F9              <1>       sub    ecx, edi
8544 0000A48C 7203              <1>       jb     short set_env_change_variable_calc14
8545 0000A48E 41                <1>       inc    ecx ; 1-512 (1-4096)
8546 0000A48F F3AA              <1>       rep    stosb ; al = 0
8547                            <1>
8548                            <1> set_env_change_variable_calc14:
8549 0000A491 29C0              <1>       sub    eax, eax ; Start address of the variable
8550                            <1>       ; EAX = 0 -> Variable is removed
8551                            <1>       ; EDX = Variable length = 0
8552                            <1>
8553 0000A493 E95FFEFFFF        <1>       jmp   set_env_string_allocate_envb_retn ; OK !
8554                            <1>
8555                            <1> set_env_change_variable_calc15:
8556 0000A498 52                <1>       push   edx ; *****
8557 0000A499 F7DA              <1>       neg    edx
8558 0000A49B 01CA              <1>       add    edx, ecx ; difference (the old string is longer)
8559 0000A49D 89F3              <1>       mov    ebx, esi
8560 0000A49F 89FE              <1>       mov    esi, edi
8561                            <1>
8562                            <1> set_env_change_variable_calc16:
8563 0000A4A1 AC                <1>       lodsb
8564 0000A4A2 20C0              <1>       and    al, al
8565 0000A4A4 75FB              <1>       jnz    short set_env_change_variable_calc16
8566                            <1>
8567 0000A4A6 B900320900        <1>       mov    ecx, Env_Page + Env_Page_Size
8568                            <1>
8569 0000A4AB 39CE              <1>       cmp    esi, ecx ; +512 (+4096)
8570 0000A4AD 7605              <1>       jna    short set_env_change_variable_calc17
8571                            <1>
8572 0000A4AF 89CE              <1>       mov    esi, ecx
8573 0000A4B1 8846FF            <1>       mov    [esi-1], al ; 0
8574                            <1>
8575                            <1> set_env_change_variable_calc17:
8576 0000A4B4 89F9              <1>       mov    ecx, edi  ; current (old) variable's address
8577 0000A4B6 89F7              <1>       mov    edi, esi  ; next variable's address
8578                            <1>
8579 0000A4B8 AC                <1>       lodsb
8580 0000A4B9 08C0              <1>       or     al, al
8581 0000A4BB 741D              <1>       jz     short set_env_change_variable_calc20
8582                            <1>
8583                            <1> set_env_change_variable_calc18:
8584 0000A4BD AC                <1>       lodsb
8585 0000A4BE 20C0              <1>       and    al, al
8586 0000A4C0 75FB              <1>       jnz    short set_env_change_variable_calc18
8587                            <1>
8588 0000A4C2 81FE00320900      <1>       cmp    esi, Env_Page + Env_Page_Size
8589 0000A4C8 720B              <1>       jb     short set_env_change_variable_calc19
8590 0000A4CA 740E              <1>       je     short set_env_change_variable_calc20
8591                            <1>
8592 0000A4CC BEFF310900        <1>       mov    esi, Env_Page + Env_Page_Size - 1
8593 0000A4D1 8806              <1>       mov    [esi], al ; 0
8594 0000A4D3 EB06              <1>       jmp    short set_env_change_variable_calc21
8595                            <1>
8596                            <1> set_env_change_variable_calc19:
8597 0000A4D5 AC                <1>       lodsb
8598 0000A4D6 08C0              <1>       or     al, al
8599 0000A4D8 75E3              <1>       jnz    short set_env_change_variable_calc18
8600                            <1>
8601                            <1> set_env_change_variable_calc20:
8602 0000A4DA 4E                <1>       dec    esi ; address of the second (last) 0 of the 00
8603                            <1>
8604                            <1> set_env_change_variable_calc21:
8605                            <1>       ; edx = difference (byte count)
8606                            <1>
8607 0000A4DB 89C8              <1>       mov    eax, ecx ; old variable's address (after '=')
8608                            <1>
8609 0000A4DD 89F1              <1>       mov    ecx, esi
8610 0000A4DF 29F9              <1>       sub    ecx, edi ; count of bytes to move backward
8611                            <1>
8612 0000A4E1 89FE              <1>       mov    esi, edi ; next variable's address
8613 0000A4E3 29D7              <1>       sub    edi, edx ; (displacement)
8614                            <1>
8615 0000A4E5 F3A4              <1>       rep    movsb
8616                            <1>
```

```
8617 0000A4E7 880F                <1>       mov   [edi], cl ; 0 ; 00 ; end of environment variables
8618                              <1>
8619 0000A4E9 89C7                <1>       mov   edi, eax
8620 0000A4EB 5A                  <1>       pop   edx ; ***** ; byte count (after '=')
8621 0000A4EC 89D1                <1>       mov   ecx, edx
8622 0000A4EE 89DE                <1>       mov   esi, ebx ; ASCIIZ string address (after '=')
8623 0000A4F0 89FB                <1>       mov   ebx, edi
8624                              <1>
8625 0000A4F2 F3A4                <1>       rep   movsb
8626                              <1>
8627 0000A4F4 880F                <1>       mov   [edi], cl ; 0 ; end of variable
8628                              <1>
8629 0000A4F6 0FB605[D88C0100]    <1>       movzx eax, byte [env_var_length]
8630 0000A4FD 01C2                <1>       add   edx, eax ; variable length (total)
8631 0000A4FF F7D8                <1>       neg   eax
8632 0000A501 01D8                <1>       add   eax, ebx ; start address of the variable
8633 0000A503 F8                  <1>       clc   ; 13/04/2016
8634 0000A504 E9EEFDFFFF          <1>       jmp   set_env_string_allocate_envb_retn ; OK !
8635                              <1>
8636                              <1> mainprog_startup_configuration:
8637                              <1>       ; 22/11/2017
8638                              <1>       ; 06/05/2016
8639                              <1>       ; 14/04/2016 (TRDOS 386 = TRDOS v2.0)
8640                              <1>       ; 17/09/2011 (TRDOS v1, MAINPROG.ASM)
8641                              <1>       ;
8642                              <1> loc_load_mainprog_cfg_file:
8643 0000A509 BE[9B390100]        <1>       mov   esi, MainProgCfgFile
8644 0000A50E 66B80018            <1>       mov   ax, 1800h ; Except volume label and dirs
8645 0000A512 E83EEAFFFF          <1>       call  find_first_file
8646 0000A517 7256                <1>       jc    short loc_load_mainprog_cfg_exit
8647                              <1>
8648                              <1>       ;or   eax, eax
8649                              <1>       ;jz   short loc_load_mainprog_cfg_exit
8650                              <1>
8651                              <1> loc_start_mainprog_configuration:
8652                              <1>       ; ESI = FindFile_DirEntry Location
8653                              <1>       ; EAX = File Size
8654                              <1>
8655 0000A519 A3[5C810100]        <1>       mov   [MainProgCfg_FileSize], eax
8656                              <1>
8657 0000A51E 668B5614            <1>       mov   dx, [esi+DirEntry_FstClusHI]
8658 0000A522 C1E210              <1>       shl   edx, 16
8659 0000A525 668B561A            <1>       mov   dx, [esi+DirEntry_FstClusLO]
8660 0000A529 8915[8C8C0100]      <1>       mov   [csftdf_sf_cluster], edx
8661                              <1>
8662 0000A52F 89C1                <1>       mov   ecx, eax
8663 0000A531 29C0                <1>       sub   eax, eax
8664                              <1>
8665                              <1>       ; TRDOS 386 (TRDOS v2.0)
8666                              <1>       ; Allocate contiguous memory block for loading the file
8667                              <1>
8668                              <1>       ; eax = 0 (Allocate memory from the beginning)
8669                              <1>       ; ecx = File (Allocation) size in bytes
8670                              <1>
8671 0000A533 E85FBAFFFF          <1>       call  allocate_memory_block
8672 0000A538 7235                <1>       jc    short loc_load_mainprog_cfg_exit
8673                              <1>
8674 0000A53A A3[848C0100]        <1>       mov   [csftdf_sf_mem_addr], eax ; loading address
8675 0000A53F 890D[888C0100]      <1>       mov   [csftdf_sf_mem_bsize], ecx ; block size
8676                              <1>
8677 0000A545 31DB                <1>       xor   ebx, ebx
8678                              <1>       ;mov  [csftdf_sf_rbytes], ebx ; 0, reset
8679                              <1>
8680 0000A547 8A3D[6E810100]      <1>       mov   bh, [Current_Drv] ; [FindFile_Drv]
8681 0000A54D BE00010900          <1>       mov   esi, Logical_DOSDisks
8682 0000A552 01DE                <1>       add   esi, ebx
8683                              <1>
8684 0000A554 8B1D[848C0100]      <1>       mov   ebx, [csftdf_sf_mem_addr] ; memory block address
8685                              <1>
8686 0000A55A 807E0300            <1>       cmp   byte [esi+LD_FATType], 0
8687 0000A55E 7710                <1>       ja    short loc_mcfg_load_fat_file
8688                              <1>
8689 0000A560 C705[948C0100]0000- <1>       mov   dword [csftdf_r_size], 65536
8689 0000A568 0100                <1>
8690 0000A56A E9A1010000          <1>       jmp   loc_mcfg_load_fs_file
8691                              <1>
8692                              <1> loc_load_mainprog_cfg_exit:
8693 0000A56F C3                  <1>       retn
8694                              <1>
8695                              <1> loc_mcfg_load_fat_file:
8696 0000A570 0FB74611            <1>       movzx eax, word [esi+LD_BPB+BytesPerSec]
8697 0000A574 0FB64E13            <1>       movzx ecx, byte [esi+LD_BPB+SecPerClust]
8698 0000A578 F7E1                <1>       mul   ecx
8699 0000A57A A3[948C0100]        <1>       mov   [csftdf_r_size], eax
8700                              <1>
8701                              <1> loc_mcfg_load_fat_file_next:
8702 0000A57F E822010000          <1>       call  mcfg_read_fat_file_sectors
8703 0000A584 0F8206010000        <1>       jc    mcfg_deallocate_mem
8704                              <1>
8705 0000A58A 09D2                <1>       or    edx, edx ; edx > 0 -> EOF
8706 0000A58C 74F1                <1>       jz    short loc_mcfg_load_fat_file_next
8707                              <1>
8708                              <1> loc_mcfg_load_fat_file_ok:
8709                              <1>       ; 06/05/2016
8710 0000A58E C705[288D0100]-     <1>       mov   dword [mainprog_return_addr], loc_mcfg_ci_return_addr
8710 0000A594 [51A60000]          <1>
8711                              <1>       ;
8712 0000A598 8B35[848C0100]      <1>       mov   esi, [csftdf_sf_mem_addr]
8713 0000A59E 8935[60810100]      <1>       mov   [MainProgCfg_LineOffset], esi
8714                              <1>
8715 0000A5A4 A1[5C810100]        <1>       mov   eax, [MainProgCfg_FileSize]
8716 0000A5A9 89C2                <1>       mov   edx, eax
8717 0000A5AB 01F2                <1>       add   edx, esi
8718                              <1>
8719                              <1> loc_mcfg_process_next_line_check:
```

```
8720 0000A5AD 89C1              <1>       mov    ecx, eax
8721                            <1>
8722 0000A5AF 803E2A            <1>       cmp    byte [esi], "*" ; Remark sign
8723 0000A5B2 7503              <1>       jne    short loc_mcfg_process_next_line
8724 0000A5B4 46                <1>       inc    esi
8725 0000A5B5 EB17              <1>       jmp    short loc_move_mainprog_cfg_nl1
8726                            <1>
8727                            <1> loc_mcfg_process_next_line:
8728 0000A5B7 83F94F            <1>       cmp    ecx, 79
8729 0000A5BA 7605              <1>       jna    short loc_start_mainprog_cfg_process
8730                            <1>
8731 0000A5BC B94F000000        <1>       mov    ecx, 79
8732                            <1>
8733                            <1> loc_start_mainprog_cfg_process:
8734 0000A5C1 BF[1E820100]      <1>       mov    edi, CommandBuffer
8735                            <1>
8736                            <1> loc_move_mainprog_cfg_line:
8737 0000A5C6 AC                <1>       lodsb
8738 0000A5C7 3C20              <1>       cmp    al, 20h
8739 0000A5C9 720C              <1>       jb     short loc_move_mainprog_cfg_nl2
8740 0000A5CB AA                <1>       stosb
8741 0000A5CC E2F8              <1>       loop   loc_move_mainprog_cfg_line
8742                            <1>
8743                            <1> loc_move_mainprog_cfg_nl1:
8744 0000A5CE 39D6              <1>       cmp    esi, edx ; + configuration file size
8745 0000A5D0 7312              <1>       jnb    short loc_end_of_mainprog_cfg_line
8746 0000A5D2 AC                <1>       lodsb
8747 0000A5D3 3C20              <1>       cmp    al, 20h
8748 0000A5D5 73F7              <1>       jnb    short loc_move_mainprog_cfg_nl1
8749                            <1>
8750                            <1> loc_move_mainprog_cfg_nl2:
8751 0000A5D7 39D6              <1>       cmp    esi, edx
8752 0000A5D9 7309              <1>       jnb    short loc_end_of_mainprog_cfg_line
8753 0000A5DB 8A06              <1>       mov    al, [esi]
8754 0000A5DD 3C20              <1>       cmp    al, 20h
8755 0000A5DF 7703              <1>       ja     short loc_end_of_mainprog_cfg_line
8756 0000A5E1 46                <1>       inc    esi
8757 0000A5E2 EBF3              <1>       jmp    short loc_move_mainprog_cfg_nl2
8758                            <1>
8759                            <1> loc_end_of_mainprog_cfg_line:
8760 0000A5E4 C60700            <1>       mov    byte [edi], 0
8761                            <1>
8762 0000A5E7 8935[60810100]    <1>       mov    [MainProgCfg_LineOffset], esi
8763                            <1>
8764                            <1>       ; 22/11/2017
8765 0000A5ED BE[26820100]      <1>       mov    esi, CommandBuffer + 8
8766 0000A5F2 29FE              <1>       sub    esi, edi
8767 0000A5F4 7606              <1>       jna    short loc_move_mainprog_cfg_command
8768 0000A5F6 30C0              <1>       xor    al, al
8769                            <1> loc_mainprog_cfg_clear_chrs:
8770 0000A5F8 AA                <1>       stosb
8771 0000A5F9 4E                <1>       dec    esi
8772 0000A5FA 75FC              <1>       jnz    short loc_mainprog_cfg_clear_chrs
8773                            <1>
8774                            <1> loc_move_mainprog_cfg_command:
8775 0000A5FC BE[1E820100]      <1>       mov    esi, CommandBuffer
8776 0000A601 89F7              <1>       mov    edi, esi
8777 0000A603 31DB              <1>       xor    ebx, ebx
8778                            <1>       ;xor   ecx, ecx
8779 0000A605 30C9              <1>       xor    cl, cl
8780                            <1>
8781                            <1> loc_move_mcfg_first_cmd_char:
8782 0000A607 8A041E            <1>       mov    al, [esi+ebx]
8783 0000A60A FEC3              <1>       inc    bl
8784 0000A60C 3C20              <1>       cmp    al, 20h
8785 0000A60E 7712              <1>       ja     short loc_move_mcfg_cmd_capitalizing
8786 0000A610 7237              <1>       jb     short loc_move_mcfg_cmd_arguments_ok
8787 0000A612 80FB4F            <1>       cmp    bl, 79
8788 0000A615 72F0              <1>       jb     short loc_move_mcfg_first_cmd_char
8789 0000A617 EB30              <1>       jmp    short loc_move_mcfg_cmd_arguments_ok
8790                            <1>
8791                            <1> loc_move_mcfg_next_cmd_char:
8792 0000A619 8A041E            <1>       mov    al, [esi+ebx]
8793 0000A61C FEC3              <1>       inc    bl
8794 0000A61E 3C20              <1>       cmp    al, 20h
8795 0000A620 7614              <1>       jna    short loc_move_mcfg_cmd_ok
8796                            <1>
8797                            <1> loc_move_mcfg_cmd_capitalizing:
8798 0000A622 3C61              <1>       cmp    al, 61h ; 'a'
8799 0000A624 7206              <1>       jb     short loc_move_mcfg_cmd_caps_ok
8800 0000A626 3C7A              <1>       cmp    al, 7Ah ; 'z'
8801 0000A628 7702              <1>       ja     short loc_move_mcfg_cmd_caps_ok
8802 0000A62A 24DF              <1>       and    al, 0DFh ; sub      al, 'a'-'A'
8803                            <1>
8804                            <1> loc_move_mcfg_cmd_caps_ok:
8805 0000A62C AA                <1>       stosb
8806 0000A62D FEC1              <1>       inc    cl
8807 0000A62F 80FB4F            <1>       cmp    bl, 79
8808 0000A632 72E5              <1>       jb     short loc_move_mcfg_next_cmd_char
8809 0000A634 EB13              <1>       jmp    short loc_move_mcfg_cmd_arguments_ok
8810                            <1>
8811                            <1> loc_move_mcfg_cmd_ok:
8812 0000A636 30C0              <1>       xor    al, al ; 0
8813                            <1>
8814                            <1> loc_move_mcfg_cmd_arguments:
8815 0000A638 8807              <1>       mov    [edi], al
8816 0000A63A 47                <1>       inc    edi
8817 0000A63B 80FB4F            <1>       cmp    bl, 79
8818 0000A63E 7309              <1>       jnb    short loc_move_mcfg_cmd_arguments_ok
8819 0000A640 8A041E            <1>       mov    al, [esi+ebx]
8820 0000A643 FEC3              <1>       inc    bl
8821 0000A645 3C20              <1>       cmp    al, 20h
8822 0000A647 73EF              <1>       jnb    short loc_move_mcfg_cmd_arguments
8823                            <1>
8824                            <1> loc_move_mcfg_cmd_arguments_ok:
```

```
8825 0000A649 C60700              <1>       mov   byte [edi], 0
8826                              <1>
8827                              <1> loc_mcfg_process_cmd_interpreter:
8828 0000A64C E825E0FFFF          <1>       call    command_interpreter
8829                              <1>
8830                              <1> loc_mcfg_ci_return_addr:
8831 0000A651 A1[5C810100]        <1>       mov   eax, [MainProgCfg_FileSize]
8832 0000A656 89C2                <1>       mov   edx, eax
8833 0000A658 8B35[60810100]      <1>       mov   esi, [MainProgCfg_LineOffset]
8834 0000A65E 01F2                <1>       add   edx, esi
8835 0000A660 0305[848C0100]      <1>       add   eax, [csftdf_sf_mem_addr]
8836 0000A666 29F0                <1>       sub   eax, esi
8837 0000A668 0F873FFFFFFF        <1>       ja    loc_mcfg_process_next_line_check
8838                              <1>
8839 0000A66E E81D000000          <1>       call  mcfg_deallocate_mem
8840                              <1>
8841 0000A673 B94F000000          <1>       mov   ecx, 79 ; 80 ?
8842 0000A678 BF[1E820100]        <1>       mov   edi, CommandBuffer
8843 0000A67D 30C0                <1>       xor   al, al
8844 0000A67F F3AA                <1>       rep   stosb
8845                              <1>
8846                              <1>       ; 06/05/2016
8847 0000A681 BE[D7430100]        <1>       mov   esi, nextline
8848 0000A686 E8BAC9FFFF          <1>       call  print_msg
8849 0000A68B E963D6FFFF          <1>       jmp   dos_prompt
8850                              <1>
8851                              <1> mcfg_deallocate_mem:
8852 0000A690 A1[848C0100]        <1>       mov   eax, [csftdf_sf_mem_addr] ; start address
8853 0000A695 8B0D[888C0100]      <1>       mov   ecx, [csftdf_sf_mem_bsize] ; block size
8854                              <1>       ;call  deallocate_memory_block
8855                              <1>       ;retn
8856 0000A69B E904BBFFFF          <1>       jmp   deallocate_memory_block
8857                              <1>
8858                              <1> mcfg_read_file_sectors:
8859                              <1>       ; 14/04/2016
8860 0000A6A0 807E0300            <1>       cmp   byte [esi+LD_FATType], 0
8861 0000A6A4 7669                <1>         jna  short mcfg_read_fs_file_sectors
8862                              <1>
8863                              <1> mcfg_read_fat_file_sectors:
8864                              <1>       ; return:
8865                              <1>       ;   CF = 0 & EDX > 0 -> END OF FILE
8866                              <1>       ;   CF = 0 & EDX = 0 -> not EOF
8867                              <1>       ;   CF = 1 -> read error (error code in AL)
8868                              <1>
8869                              <1> mcfg_read_fat_file_secs_0:
8870 0000A6A6 8B15[5C810100]      <1>       mov   edx, [MainProgCfg_FileSize]
8871 0000A6AC 2B15[9C8C0100]      <1>       sub   edx, [csftdf_sf_rbytes]
8872 0000A6B2 3B15[948C0100]      <1>       cmp   edx, [csftdf_r_size]
8873 0000A6B8 7306                <1>       jnb   short mcfg_read_fat_file_secs_1
8874 0000A6BA 8915[948C0100]      <1>       mov   [csftdf_r_size], edx
8875                              <1>
8876                              <1> mcfg_read_fat_file_secs_1:
8877 0000A6C0 A1[948C0100]        <1>       mov   eax, [csftdf_r_size]
8878 0000A6C5 29D2                <1>       sub   edx, edx
8879 0000A6C7 0FB74E11            <1>       movzx ecx, word [esi+LD_BPB+BytesPerSec]
8880 0000A6CB 01C8                <1>       add   eax, ecx
8881 0000A6CD 48                  <1>       dec   eax
8882 0000A6CE F7F1                <1>       div   ecx
8883 0000A6D0 89C1                <1>       mov   ecx, eax ; sector count
8884 0000A6D2 A1[8C8C0100]        <1>       mov   eax, [csftdf_sf_cluster]
8885                              <1>
8886                              <1>       ; EBX = memory block address (current)
8887                              <1>
8888 0000A6D7 E88C230000          <1>       call  read_fat_file_sectors
8889 0000A6DC 7230                <1>       jc    short mcfg_read_fat_file_secs_3
8890                              <1>
8891                              <1>       ; EBX = next memory address
8892                              <1>
8893 0000A6DE A1[9C8C0100]        <1>       mov   eax, [csftdf_sf_rbytes]
8894 0000A6E3 0305[948C0100]      <1>       add   eax, [csftdf_r_size]
8895 0000A6E9 8B15[5C810100]      <1>       mov   edx, [MainProgCfg_FileSize]
8896 0000A6EF 39D0                <1>       cmp   eax, edx
8897 0000A6F1 731B                <1>       jnb   short mcfg_read_fat_file_secs_3 ; edx > 0
8898 0000A6F3 A3[9C8C0100]        <1>       mov   [csftdf_sf_rbytes], eax
8899                              <1>
8900 0000A6F8 53                  <1>       push  ebx ; *
8901                              <1>       ; get next cluster (csftdf_r_size! bytes)
8902 0000A6F9 A1[8C8C0100]        <1>       mov   eax, [csftdf_sf_cluster]
8903 0000A6FE E837210000          <1>       call  get_next_cluster
8904 0000A703 5B                  <1>       pop   ebx ; *
8905 0000A704 7301                <1>       jnc   short mcfg_read_fat_file_secs_2
8906                              <1>
8907                              <1>       ;mov  eax, 17; Read error !
8908 0000A706 C3                  <1>       retn
8909                              <1>
8910                              <1> mcfg_read_fat_file_secs_2:
8911 0000A707 29D2                <1>       sub   edx, edx ; 0
8912 0000A709 A3[8C8C0100]        <1>       mov   [csftdf_sf_cluster], eax ; next cluster
8913                              <1>
8914                              <1> mcfg_read_fat_file_secs_3:
8915 0000A70E C3                  <1>       retn
8916                              <1>
8917                              <1> mcfg_read_fs_file_sectors:
8918 0000A70F C3                  <1>       retn
8919                              <1>
8920                              <1> loc_mcfg_load_fs_file:
8921 0000A710 C3                  <1>       retn
8922                              <1>
8923                              <1> load_and_execute_file:
8924                              <1>       ; 04/01/2017
8925                              <1>       ; 06/05/2016, 07/05/2016, 11/05/2016
8926                              <1>       ; 23/04/2016, 24/04/2016
8927                              <1>       ; 22/04/2016 (TRDOS 386 = TRDOS v2.0)
8928                              <1>       ; 05/11/2011
8929                              <1>       ; (TRDOS v1, CMDINTR.ASM, 'cmp_cmd_run', 'cmp_cmd_external')
```

```
8930                            <1>       ; ('loc_run_check_filename')
8931                            <1>       ; 29/08/2011
8932                            <1>       ; 10/09/2011
8933                            <1>       ; INPUT->
8934                            <1>       ;     ESI = Path Name address (CommandBuffer address)
8935                            <1>       ; OUTPUT ->
8936                            <1>       ;     none (error message will be shown if an error will occur)
8937                            <1>       ;
8938                            <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI, EBP will be changed)
8939                            <1>       ;
8940                            <1> loc_run_check_filename:
8941 0000A711 803E20             <1>       cmp    byte [esi], 20h
8942 0000A714 0F822BE3FFFF       <1>       jb     loc_cmd_failed
8943 0000A71A 7703               <1>       ja     short loc_run_check_filename_ok
8944 0000A71C 46                 <1>       inc    esi
8945 0000A71D EBF2               <1>       jmp    short loc_run_check_filename
8946                            <1>
8947                            <1> loc_run_check_filename_ok:
8948 0000A71F C605[CF810100]00   <1>       mov    byte [CmdArgStart], 0 ; reset
8949 0000A726 56                 <1>       push   esi ; *
8950                            <1> loc_run_get_first_arg_pos:
8951 0000A727 46                 <1>       inc    esi
8952 0000A728 8A06               <1>       mov    al, [esi]
8953 0000A72A 3C20               <1>       cmp    al, 20h
8954 0000A72C 77F9               <1>       ja     short loc_run_get_first_arg_pos
8955 0000A72E C60600             <1>       mov    byte [esi], 0
8956                            <1> loc_run_get_external_arg_pos:
8957                            <1>       ; 11/05/2016
8958 0000A731 46                 <1>       inc    esi
8959 0000A732 8A06               <1>       mov    al, [esi]
8960 0000A734 3C20               <1>       cmp    al, 20h
8961 0000A736 760C               <1>       jna    short loc_run_parse_path_name
8962 0000A738 89F0               <1>       mov    eax, esi
8963 0000A73A 2D[1E820100]       <1>       sub    eax, CommandBuffer
8964 0000A73F A2[CF810100]       <1>       mov    byte [CmdArgStart], al
8965                            <1> loc_run_parse_path_name:
8966 0000A744 5E                 <1>       pop    esi ; *
8967 0000A745 BF[0E8A0100]       <1>       mov    edi, FindFile_Drv
8968 0000A74A E8D7090000         <1>       call   parse_path_name
8969 0000A74F 0F82F0E2FFFF       <1>       jc     loc_cmd_failed
8970                            <1>
8971                            <1> loc_run_check_filename_exists:
8972 0000A755 BE[508A0100]       <1>       mov    esi, FindFile_Name
8973 0000A75A 803E20             <1>       cmp    byte [esi], 20h
8974 0000A75D 0F86E2E2FFFF       <1>       jna    loc_cmd_failed
8975                            <1>
8976                            <1> loc_run_check_exe_filename_ext:
8977 0000A763 E890020000         <1>       call   check_prg_filename_ext
8978 0000A768 0F82D7E2FFFF       <1>       jc     loc_cmd_failed
8979                            <1>
8980                            <1> loc_run_check_exe_filename_ext_ok:
8981 0000A76E 66A3[268D0100]     <1>       mov    word [EXE_ID], ax
8982                            <1>
8983                            <1> loc_run_drv:
8984 0000A774 C605[258D0100]00   <1>       mov    byte [Run_Manual_Path], 0
8985 0000A77B A1[68810100]       <1>       mov    eax, [Current_Dir_FCluster]
8986 0000A780 A3[208D0100]       <1>       mov   [Run_CDirFC], eax
8987                            <1>       ;
8988 0000A785 8A35[6E810100]     <1>       mov    dh, [Current_Drv]
8989 0000A78B 8835[CA880100]     <1>       mov    [RUN_CDRV], dh
8990                            <1>
8991 0000A791 8A15[0E8A0100]     <1>       mov    dl, [FindFile_Drv]
8992 0000A797 38F2               <1>       cmp    dl, dh
8993 0000A799 7412               <1>       je     short loc_run_change_directory
8994                            <1>
8995 0000A79B 8005[258D0100]02   <1>       add    byte [Run_Manual_Path], 2
8996                            <1>
8997 0000A7A2 E80BD4FFFF         <1>       call   change_current_drive
8998 0000A7A7 0F82C3E2FFFF       <1>       jc     loc_run_cmd_failed
8999                            <1>
9000                            <1> loc_run_change_directory:
9001 0000A7AD 803D[0F8A0100]20   <1>       cmp    byte [FindFile_Directory], 20h
9002 0000A7B4 7623               <1>       jna    short loc_run_find_executable_file
9003                            <1>
9004 0000A7B6 FE05[258D0100]     <1>       inc    byte [Run_Manual_Path]
9005                            <1>
9006 0000A7BC FE05[55390100]     <1>       inc    byte [Restore_CDIR]
9007                            <1>
9008 0000A7C2 BE[0F8A0100]       <1>       mov    esi, FindFile_Directory
9009 0000A7C7 30E4               <1>       xor    ah, ah ; CD_COMMAND sign -> 0
9010 0000A7C9 E842030000         <1>       call   change_current_directory
9011 0000A7CE 0F829CE2FFFF       <1>       jc     loc_run_cmd_failed
9012                            <1>
9013                            <1> loc_run_change_prompt_dir_string:
9014 0000A7D4 E857020000         <1>       call   change_prompt_dir_string
9015                            <1>
9016                            <1> loc_run_find_executable_file:
9017 0000A7D9 66C705[248D0100]00- <1>      mov    word [Run_Auto_Path], 0
9017 0000A7E1 00                 <1>
9018                            <1>
9019                            <1> loc_run_find_executable_file_next:
9020 0000A7E2 BE[508A0100]       <1>       mov    esi, FindFile_Name
9021                            <1> loc_run_find_program_file_next:
9022 0000A7E7 66B80018           <1>       mov    ax, 1800h ; Except volume label and dirs
9023 0000A7EB E865E7FFFF         <1>       call   find_first_file
9024                            <1>       ; ESI = Directory Entry (FindFile_DirEntry) Location
9025                            <1>       ; EDI = Directory Buffer Directory Entry Location
9026                            <1>       ; EAX = File size
9027 0000A7F0 0F835C010000       <1>       jnc    loc_load_and_run_file
9028                            <1>
9029 0000A7F6 3C02               <1>       cmp    al, 2 ; file not found
9030 0000A7F8 0F8572E2FFFF       <1>       jne    loc_run_cmd_failed
9031                            <1>
9032 0000A7FE 66A1[268D0100]     <1>       mov    ax, word [EXE_ID]
9033 0000A804 80FC2E             <1>       cmp    ah, '.' ; File name has extension sign
```

```
9034 0000A807 7424                <1>       je    short loc_run_check_auto_path
9035                              <1>
9036 0000A809 08C0                <1>       or    al, al
9037 0000A80B 7520                <1>       jnz   short loc_run_check_auto_path
9038                              <1>
9039 0000A80D 80FC08              <1>       cmp   ah, 8 ; count of file name chars
9040 0000A810 771B                <1>       ja    short loc_run_check_auto_path
9041                              <1>
9042                              <1> loc_run_change_file_ext_to_prg:
9043 0000A812 0FB6DC              <1>       movzx ebx, ah ; count of file name chars
9044 0000A815 BE[508A0100]        <1>       mov   esi, FindFile_Name
9045 0000A81A 01F3                <1>       add   ebx, esi
9046                              <1>       ; 07/05/2016
9047 0000A81C C7032E505247        <1>       mov   dword [ebx],  '.PRG'
9048 0000A822 66C705[268D0100]50- <1>       mov   word [EXE_ID], 'P.'
9048 0000A82A 2E                  <1>
9049 0000A82B EBBA                <1>       jmp   short loc_run_find_program_file_next
9050                              <1>
9051                              <1> loc_run_check_auto_path:
9052                              <1>       ; NOTE: /// 07/05/2016 ///
9053                              <1>       ; If the path is given, value of byte [Run_Manual_Path]
9054                              <1>       ; will not be ZERO. If so, file searching by using
9055                              <1>       ; Automatic Path (via 'PATH' environment variable)
9056                              <1>       ; will not be applicable, because the program file
9057                              <1>       ; is already/absolutely not found.
9058                              <1>
9059 0000A82D A0[258D0100]        <1>       mov   al, [Run_Manual_Path]
9060 0000A832 08C0                <1>       or    al, al
9061 0000A834 0F850BE2FFFF        <1>       jnz   loc_cmd_failed
9062                              <1>
9063                              <1> loc_run_check_auto_path_again:
9064 0000A83A 66833D[248D0100]FF  <1>       cmp   word [Run_Auto_Path], 0FFFFh
9065                              <1>             ; 0FFFFh = Not a valid run path (in ENV block)
9066 0000A842 0F83FDE1FFFF        <1>       jnb   loc_cmd_failed
9067                              <1>       ; xor  al, al
9068 0000A848 BE[213A0100]        <1>       mov   esi, Cmd_Path ; 'PATH'
9069 0000A84D BF[6E820100]        <1>       mov   edi, TextBuffer
9070 0000A852 E848F9FFFF          <1>       call  get_environment_string
9071 0000A857 730E                <1>       jnc   short loc_run_chk_filename_ext_again
9072 0000A859 66C705[248D0100]FF- <1>       mov   word [Run_Auto_Path], 0FFFFh ; invalid
9072 0000A861 FF                  <1>
9073 0000A862 E9DEE1FFFF          <1>       jmp   loc_cmd_failed
9074                              <1>
9075                              <1> loc_run_chk_filename_ext_again:
9076 0000A867 89C1                <1>       mov   ecx, eax ; string length (with zero tail)
9077 0000A869 49                  <1>       dec   ecx ; without zero tail
9078 0000A86A 66A1[268D0100]      <1>       mov   ax, [EXE_ID]
9079 0000A870 80FC2E              <1>       cmp   ah, '.'
9080 0000A873 740E                <1>       je    short loc_run_chk_auto_path_pos
9081                              <1>
9082                              <1> loc_run_change_file_ext_to_noext_again:
9083 0000A875 0FB6DC              <1>       movzx ebx, ah
9084 0000A878 BE[508A0100]        <1>       mov   esi, FindFile_Name
9085 0000A87D 01F3                <1>       add   ebx, esi
9086 0000A87F 29C0                <1>       sub   eax, eax
9087 0000A881 8903                <1>       mov   [ebx], eax ; 0 ; erase extension (.PRG)
9088                              <1>
9089                              <1> loc_run_chk_auto_path_pos:
9090                              <1>       ;movzx eax,  word [Run_Auto_Path]
9091 0000A883 66A1[248D0100]      <1>       mov   ax, [Run_Auto_Path]
9092 0000A889 39C8                <1>       cmp   eax, ecx ; ecx = string length (except zero tail)
9093 0000A88B 0F83B4E1FFFF        <1>       jnb   loc_cmd_failed
9094                              <1>       ;or   eax, eax
9095 0000A891 6609C0              <1>       or    ax, ax
9096 0000A894 7502                <1>       jnz   short loc_run_auto_path_pos_move
9097 0000A896 B005                <1>       mov   al, 5
9098                              <1>
9099                              <1> loc_run_auto_path_pos_move:
9100 0000A898 89FE                <1>       mov   esi, edi ; offset TextBuffer
9101 0000A89A 01C6                <1>       add   esi, eax
9102                              <1>
9103                              <1> loc_run_auto_path_pos_space_loop:
9104 0000A89C AC                  <1>       lodsb
9105 0000A89D 3C20                <1>       cmp   al, 20h
9106 0000A89F 74FB                <1>       je    short loc_run_auto_path_pos_space_loop
9107 0000A8A1 0F829EE1FFFF        <1>       jb    loc_cmd_failed
9108 0000A8A7 AA                  <1>       stosb
9109                              <1> loc_run_auto_path_pos_move_next:
9110 0000A8A8 AC                  <1>       lodsb
9111 0000A8A9 3C3B                <1>       cmp   al, ';'
9112 0000A8AB 7414                <1>       je    short loc_run_auto_path_pos_move_last_byte
9113 0000A8AD 3C20                <1>       cmp   al, 20h
9114 0000A8AF 74F7                <1>       je    short loc_run_auto_path_pos_move_next
9115 0000A8B1 7203                <1>       jb    short loc_byte_ptr_end_of_path
9116 0000A8B3 AA                  <1>       stosb
9117 0000A8B4 EBF2                <1>       jmp   short loc_run_auto_path_pos_move_next
9118                              <1>
9119                              <1> loc_byte_ptr_end_of_path:
9120 0000A8B6 66C705[248D0100]FF- <1>       mov   word [Run_Auto_Path], 0FFFFh ; end of path
9120 0000A8BE FF                  <1>
9121 0000A8BF EB0D                <1>       jmp   short loc_run_auto_path_move_ok
9122                              <1>
9123                              <1> loc_run_auto_path_pos_move_last_byte:
9124 0000A8C1 89F0                <1>       mov   eax, esi
9125 0000A8C3 2D[6E820100]        <1>       sub   eax, TextBuffer
9126 0000A8C8 66A3[248D0100]      <1>       mov   [Run_Auto_Path], ax ; next path position
9127                              <1>
9128                              <1> loc_run_auto_path_move_ok:
9129 0000A8CE 4F                  <1>       dec   edi
9130 0000A8CF B02F                <1>       mov   al, '/'
9131 0000A8D1 3807                <1>       cmp   [edi], al
9132 0000A8D3 7403                <1>       je    short loc_run_auto_path_move_file_name
9133 0000A8D5 47                  <1>       inc   edi
9134 0000A8D6 8807                <1>       mov   [edi], al
9135                              <1>
```

```
                                <1> loc_run_auto_path_move_file_name:
0000A8D8 47                     <1>         inc     edi
0000A8D9 BE[508A0100]           <1>         mov     esi, FindFile_Name
                                <1>
                                <1> loc_run_auto_path_move_fn_loop:
0000A8DE AC                     <1>         lodsb
0000A8DF AA                     <1>         stosb
0000A8E0 08C0                   <1>         or      al, al
0000A8E2 75FA                   <1>         jnz     short loc_run_auto_path_move_fn_loop
                                <1>
0000A8E4 BE[6E820100]           <1>         mov     esi, TextBuffer
0000A8E9 BF[0E8A0100]           <1>         mov     edi, FindFile_Drv
0000A8EE E833080000             <1>         call    parse_path_name
0000A8F3 0F824CE1FFFF           <1>         jc      loc_cmd_failed
                                <1>
0000A8F9 8A35[6E810100]         <1>         mov     dh, [Current_Drv]
0000A8FF 8A15[0E8A0100]         <1>         mov     dl, [FindFile_Drv]
0000A905 38F2                   <1>         cmp     dl, dh
0000A907 740B                   <1>         je      short loc_run_change_directory_again
                                <1>
0000A909 E8A4D2FFFF             <1>         call    change_current_drive
0000A90E 0F825CE1FFFF           <1>         jc      loc_run_cmd_failed
                                <1>
                                <1> loc_run_change_directory_again:
0000A914 803D[0F8A0100]20       <1>         cmp     byte [FindFile_Directory], 20h
0000A91B 761D                   <1>         jna     short loc_load_executable_cdir_chk_again
                                <1>
0000A91D FE05[55390100]         <1>         inc     byte [Restore_CDIR]
0000A923 BE[0F8A0100]           <1>         mov     esi, FindFile_Directory
0000A928 30E4                   <1>         xor     ah, ah ; CD_COMMAND sign -> 0
0000A92A E8E1010000             <1>         call    change_current_directory
0000A92F 0F823BE1FFFF           <1>         jc      loc_run_cmd_failed
                                <1>
                                <1> loc_run_chg_prompt_dir_str_again:
0000A935 E8F6000000             <1>         call    change_prompt_dir_string
                                <1>
                                <1> loc_load_executable_cdir_chk_again:
0000A93A A1[68810100]           <1>         mov     eax, [Current_Dir_FCluster]
0000A93F 3B05[208D0100]         <1>         cmp     eax, [Run_CDirFC]
0000A945 0F8597FEFFFF           <1>         jne     loc_run_find_executable_file_next
0000A94B 30C0                   <1>         xor     al, al ; 0
0000A94D E9E8FEFFFF             <1>         jmp     loc_run_check_auto_path_again
                                <1>
                                <1> loc_load_and_run_file:
                                <1>         ; 13/11/2017
                                <1>         ; 04/01/2017
                                <1>         ; 23/04/2016
0000A952 BE[508A0100]           <1>         mov     esi, FindFile_Name
0000A957 BF[6E820100]           <1>         mov     edi, TextBuffer
                                <1>
                                <1>         ; 24/04/2016
0000A95C 31D2                   <1>         xor     edx, edx
0000A95E 668915[4A040300]       <1>         mov     word [argc], dx ; 0
0000A965 8915[8C030300]         <1>         mov     dword [u.nread], edx ; 0
                                <1>
                                <1> loc_load_and_run_file_1:
0000A96B AC                     <1>         lodsb
0000A96C AA                     <1>         stosb
0000A96D FF05[8C030300]         <1>         inc     dword [u.nread]
0000A973 20C0                   <1>         and     al, al
0000A975 75F4                   <1>         jnz     short loc_load_and_run_file_1
                                <1>
0000A977 A0[CF810100]           <1>         mov     al, [CmdArgStart]
0000A97C 20C0                   <1>         and     al, al
0000A97E 7445                   <1>         jz      short loc_load_and_run_file_7
                                <1>
0000A980 0FB6F0                 <1>         movzx   esi, al ; 11/05/2016
0000A983 B950000000             <1>         mov     ecx, 80
0000A988 29F1                   <1>         sub     ecx, esi
0000A98A 81C6[1E820100]         <1>         add     esi, CommandBuffer
                                <1>
0000A990 66FF05[4A040300]       <1>         inc     word [argc] ; 11/05/2016
                                <1>
                                <1> loc_load_and_run_file_2:
0000A997 AC                     <1>         lodsb
0000A998 3C20                   <1>         cmp     al, 20h
0000A99A 7717                   <1>         ja      short loc_load_and_run_file_5
0000A99C 721E                   <1>         jb      short loc_load_and_run_file_6
                                <1>
                                <1> loc_load_and_run_file_3:
0000A99E 803E20                 <1>         cmp     byte [esi], 20h
0000A9A1 7707                   <1>         ja      short loc_load_and_run_file_4
0000A9A3 7217                   <1>         jb      short loc_load_and_run_file_6
0000A9A5 46                     <1>         inc     esi
0000A9A6 E2F6                   <1>         loop    loc_load_and_run_file_3
0000A9A8 EB12                   <1>         jmp     short loc_load_and_run_file_6
                                <1>
                                <1> loc_load_and_run_file_4:
0000A9AA 28C0                   <1>         sub     al, al ; 0
0000A9AC 66FF05[4A040300]       <1>         inc     word [argc]
                                <1> loc_load_and_run_file_5:
0000A9B3 AA                     <1>         stosb
0000A9B4 FF05[8C030300]         <1>         inc     dword [u.nread]
0000A9BA E2DB                   <1>         loop    loc_load_and_run_file_2
                                <1>
                                <1> loc_load_and_run_file_6:
0000A9BC 30C0                   <1>         xor     al, al ; 0
0000A9BE AA                     <1>         stosb
0000A9BF FF05[8C030300]         <1>         inc     dword [u.nread]
                                <1> loc_load_and_run_file_7:
0000A9C5 8807                   <1>         mov     [edi], al ; 0
0000A9C7 66FF05[4A040300]       <1>         inc     word [argc] ; 24/04/2016
0000A9CE FF05[8C030300]         <1>         inc     dword [u.nread] ; 24/04/2016
0000A9D4 BE[6E820100]           <1>         mov     esi, TextBuffer
0000A9D9 8B15[7C8A0100]         <1>         mov     edx, [FindFile_DirEntry+DirEntry_FileSize]
```

```
9241 0000A9DF 66A1[748A0100]      <1>       mov    ax, [FindFile_DirEntry+DirEntry_FstClusHI]
9242 0000A9E5 C1E010              <1>       shl    eax, 16 ; 13/11/2017
9243 0000A9E8 66A1[7A8A0100]      <1>       mov    ax, [FindFile_DirEntry+DirEntry_FstClusLO]
9244                              <1>       ; EAX = First Cluster number
9245                              <1>       ; EDX = File Size
9246                              <1>       ; ESI = Argument list address
9247                              <1>       ; [argc] = argument count
9248                              <1>       ; [u.nread] = argument list length
9249 0000A9EE E874630000          <1>       call   load_and_run_file ; trdosk6.s
9250                              <1>       ;jc    loc_run_cmd_failed ; 04/01/2017
9251                              <1> loc_load_and_run_file_8: ; 06/05/2016
9252 0000A9F3 E98BE9FFFF          <1>       jmp    loc_file_rw_restore_retn
9253                              <1>
9254                              <1> check_prg_filename_ext:
9255                              <1>       ; 23/04/2016 (TRDOS 386 = TRDOS v2.0)
9256                              <1>       ; 10/09/2011
9257                              <1>       ; (TRDOS v1, CMDINTR.ASM, 'proc_check_exe_filename_ext')
9258                              <1>       ; 14/11/2009
9259                              <1>       ; INPUT ->
9260                              <1>       ;      ESI = Dot File Name
9261                              <1>       ; OUTPUT ->
9262                              <1>       ;     cf = 0 -> EXE_ID in AL
9263                              <1>       ;      ESI = Last char + 1 position
9264                              <1>       ;     cf = 1 -> Invalid executable file name
9265                              <1>       ;      or no file name extension if AH<=8
9266                              <1>       ;      AL = Last file name char
9267                              <1>       ;     cf = 0 -> AL='P' (PRG), AL=0 (no extension)
9268                              <1>       ;
9269                              <1>       ; (Modified registers: EAX, ESI)
9270                              <1>
9271 0000A9F8 30E4                <1>       xor    ah, ah
9272                              <1> loc_run_check_filename_ext:
9273 0000A9FA AC                  <1>       lodsb
9274 0000A9FB 3C21                <1>       cmp    al, 21h
9275 0000A9FD 7229                <1>       jb     short loc_check_exe_fn_retn
9276 0000A9FF FEC4                <1>       inc    ah
9277 0000AA01 3C2E                <1>       cmp    al, '.'
9278 0000AA03 75F5                <1>       jne    short loc_run_check_filename_ext
9279                              <1>
9280                              <1> loc_run_check_filename_ext_dot:
9281 0000AA05 80FC02              <1>       cmp    ah, 2 ; .??? is not valid
9282 0000AA08 88C4                <1>       mov    ah, al ; '.'
9283 0000AA0A 7219                <1>       jb     short loc_check_prg_fn_retn
9284                              <1>
9285                              <1> loc_run_check_filename_ext_dot_ok:
9286 0000AA0C AC                  <1>       lodsb
9287 0000AA0D 24DF                <1>       and    al, 0DFh
9288                              <1>
9289                              <1> loc_run_check_filename_ext_prg:
9290 0000AA0F 3C50                <1>       cmp    al, 'P'
9291 0000AA11 7212                <1>       jb     short loc_check_prg_fn_retn
9292 0000AA13 7711                <1>       ja     short loc_check_prg_fn_stc
9293 0000AA15 AC                  <1>       lodsb
9294 0000AA16 24DF                <1>       and    al, 0DFh
9295 0000AA18 3C52                <1>       cmp    al, 'R'
9296 0000AA1A 750A                <1>       jne    short loc_check_prg_fn_stc
9297 0000AA1C AC                  <1>       lodsb
9298 0000AA1D 24DF                <1>       and    al, 0DFh
9299 0000AA1F 3C47                <1>       cmp    al, 'G'
9300 0000AA21 7503                <1>       jne    short loc_check_prg_fn_stc
9301                              <1>
9302 0000AA23 B050                <1>       mov    al, 'P'
9303                              <1> loc_check_prg_fn_retn:
9304 0000AA25 C3                  <1>       retn
9305                              <1>
9306                              <1> loc_check_prg_fn_stc:
9307 0000AA26 F9                  <1>       stc
9308 0000AA27 C3                  <1>       retn
9309                              <1>
9310                              <1> loc_check_exe_fn_retn:
9311 0000AA28 28C0                <1>       sub    al, al ; 0
9312 0000AA2A C3                  <1>       retn
9313                              <1>
9314                              <1> find_and_list_files:
9315 0000AA2B C3                  <1>       retn
9316                              <1> set_exec_arguments:
9317 0000AA2C C3                  <1>       retn
9318                              <1> delete_fs_directory:
9319 0000AA2D 31C0                <1>       xor eax, eax
9320 0000AA2F C3                  <1>       retn
3109                                  %include 'trdosk4.s' ; 24/01/2016
3110                              <1> ; ****************************************************************************
3111                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - Directory Functions : trdosk4.s
3112                              <1> ; ----------------------------------------------------------------------------
3113                              <1> ; Last Update: 02/03/2021
3114                              <1> ; ----------------------------------------------------------------------------
3115                              <1> ; Beginning: 24/01/2016
3116                              <1> ; ----------------------------------------------------------------------------
3117                              <1> ; Assembler: NASM version 2.11 (trdos386.s)
3118                              <1> ; ----------------------------------------------------------------------------
3119                              <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
3120                              <1> ; DIR.ASM (09/10/2011)
3121                              <1> ; ****************************************************************************
3122                              <1>
3123                              <1> ; DIR.ASM  [ TRDOS KERNEL - COMMAND EXECUTER SECTION - DIRECTORY FUNCTIONS ]
3124                              <1> ; (c) 2004-2010  Erdogan TAN  [ 17/01/2004 ]  Last Update: 09/10/2011
3125                              <1> ; FILE.ASM [ FILE FUNCTIONS ] Last Update: 09/10/2011
3126                              <1>
3127                              <1> change_prompt_dir_string:
3128                              <1>       ; 05/10/2016
3129                              <1>       ; 24/01/2016 (TRDOS 386 = TRDOS v2.0)
3130                              <1>       ; 27/03/2011
3131                              <1>       ; 09/10/2009
3132                              <1>       ; INPUT/OUTPUT => none
3133                              <1>       ; this procedure changes current directory string/text
```

```
3134                                    <1>       ; 2005
3135                                    <1>
3136 0000AA30 BE[CB880100]              <1>       mov    esi, PATH_Array
3137                                    <1> change_prompt_dir_str: ; 05/10/2016 (call from 'set_working_path')
3138 0000AA35 BF[72810100]              <1>       mov    edi, Current_Directory
3139 0000AA3A 8A25[6C810100]            <1>       mov    ah, [Current_Dir_Level]
3140 0000AA40 E807000000                <1>       call   set_current_directory_string
3141 0000AA45 880D[CD810100]            <1>       mov    [Current_Dir_StrLen], cl
3142                                    <1>
3143 0000AA4B C3                        <1>       retn
3144                                    <1>
3145                                    <1> set_current_directory_string:
3146                                    <1>       ; 24/01/2016 (TRDOS 386 = TRDOS v2.0)
3147                                    <1>       ; 27/03/2011
3148                                    <1>       ; 09/10/2009
3149                                    <1>       ; INPUT:
3150                                    <1>       ;    ESI = Path Array Address
3151                                    <1>       ;    EDI = Current Directory String Buffer
3152                                    <1>       ;    AH = Current Directory Level
3153                                    <1>       ; OUTPUT => EAX, EBX, ESI will be changed
3154                                    <1>       ;    EDI will be same with input
3155                                    <1>       ;    ECX = Current Directory String Length
3156                                    <1>
3157 0000AA4C 57                        <1>       push    edi
3158 0000AA4D 80FC00                    <1>       cmp     ah, 0
3159 0000AA50 7652                      <1>       jna    short pass_write_path
3160 0000AA52 83C610                    <1>       add    esi, 16
3161 0000AA55 89F3                      <1>       mov    ebx, esi
3162                                    <1> loc_write_path:
3163 0000AA57 B908000000                <1>       mov    ecx, 8
3164                                    <1> path_write_dirname1:
3165 0000AA5C AC                        <1>       lodsb
3166 0000AA5D 3C20                      <1>       cmp    al, 20h
3167 0000AA5F 7612                      <1>       jna    short pass_write_dirname1
3168 0000AA61 AA                        <1>       stosb
3169 0000AA62 81FF[CC810100]            <1>       cmp    edi, End_Of_Current_Dir_Str
3170 0000AA68 733A                      <1>       jnb    short pass_write_path
3171 0000AA6A E2F0                      <1>       loop   path_write_dirname1
3172 0000AA6C 803E20                    <1>       cmp    byte [esi], 20h
3173 0000AA6F 7624                      <1>       jna    short pass_write_dirname2
3174 0000AA71 EB0A                      <1>       jmp     short loc_put_dot_cont_ext
3175                                    <1> pass_write_dirname1:
3176 0000AA73 89DE                      <1>       mov    esi, ebx
3177 0000AA75 83C608                    <1>       add    esi, 8
3178 0000AA78 803E20                    <1>       cmp    byte [esi], 20h
3179 0000AA7B 7618                      <1>       jna    short pass_write_dirname2
3180                                    <1> loc_put_dot_cont_ext:
3181 0000AA7D C6072E                    <1>       mov    byte [edi], "."
3182                                    <1>       ;mov   ecx, 3
3183 0000AA80 B103                      <1>       mov    cl, 3
3184                                    <1> loc_check_dir_name_ext:
3185 0000AA82 AC                        <1>       lodsb
3186 0000AA83 47                        <1>       inc    edi
3187 0000AA84 3C20                      <1>       cmp    al, 20h
3188 0000AA86 760D                      <1>       jna    short pass_write_dirname2
3189 0000AA88 8807                      <1>       mov    [edi], al
3190 0000AA8A 81FF[CC810100]            <1>       cmp    edi, End_Of_Current_Dir_Str
3191 0000AA90 7312                      <1>       jnb    short pass_write_path
3192 0000AA92 E2EE                      <1>       loop    loc_check_dir_name_ext
3193 0000AA94 47                        <1>       inc    edi
3194                                    <1> pass_write_dirname2:
3195 0000AA95 FECC                      <1>       dec    ah
3196 0000AA97 740B                      <1>       jz      short pass_write_path
3197 0000AA99 83C310                    <1>       add    ebx, 16
3198 0000AA9C 89DE                      <1>       mov    esi, ebx
3199 0000AA9E C6072F                    <1>       mov    byte [edi],"/"
3200 0000AAA1 47                        <1>       inc    edi
3201 0000AAA2 EBB3                      <1>       jmp    short loc_write_path
3202                                    <1> pass_write_path:
3203 0000AAA4 C60700                    <1>       mov    byte [edi], 0
3204 0000AAA7 47                        <1>       inc    edi
3205 0000AAA8 89F9                      <1>       mov    ecx, edi
3206 0000AAAA 5F                        <1>       pop    edi
3207 0000AAAB 29F9                      <1>       sub    ecx, edi
3208                                    <1>       ; ECX = Current Directory String Length
3209 0000AAAD C3                        <1>       retn
3210                                    <1>
3211                                    <1> get_current_directory:
3212                                    <1>       ; 15/10/2016
3213                                    <1>       ; 14/02/2016
3214                                    <1>       ; 24/01/2016 (TRDOS 386 = TRDOS v2.0)
3215                                    <1>       ; 27/03/2011
3216                                    <1>       ;
3217                                    <1>       ; INPUT-> ESI = Current Directory Buffer
3218                                    <1>       ;         DL = TRDOS Logical Dos Drive Number + 1
3219                                    <1>       ;              (0= Default/Current Drive)
3220                                    <1>       ;
3221                                    <1>       ;   Note: Required dir buffer length may be <= 92 bytes
3222                                    <1>       ;         for TRDOS (7*12 name chars + 7 slash + 0)
3223                                    <1>       ; OUTPUT ->  ESI = Current Directory Buffer
3224                                    <1>       ;            EAX, EBX, ECX, EDX, EDI will be changed
3225                                    <1>       ;            CX/CL = Current Directory String Length
3226                                    <1>       ;           DL = Drive Number (0 based)
3227                                    <1>       ;             (If input is 0, output is current drv number)
3228                                    <1>       ;            DH = same with input
3229                                    <1>       ;   cf = 0 -> AL = 0
3230                                    <1>       ;   cf = 1 -> error code in AL
3231                                    <1>
3232                                    <1> loc_get_current_drive_0:
3233 0000AAAE 80FA00                    <1>       cmp    dl, 0
3234 0000AAB1 7708                      <1>       ja     short loc_get_current_drive_1
3235 0000AAB3 8A15[6E810100]            <1>       mov    dl, [Current_Drv]
3236 0000AAB9 EB17                      <1>       jmp    short loc_get_current_drive_2
3237                                    <1> loc_get_current_drive_1:
3238 0000AABB FECA                      <1>       dec    dl
```

```
3239 0000AABD 3A15[54390100]      <1>       cmp    dl, [Last_DOS_DiskNo]
3240 0000AAC3 760D                <1>       jna    short loc_get_current_drive_2
3241 0000AAC5 B80F000000          <1>       mov    eax, 0Fh ; Invalid drive (Drive not ready!)
3242 0000AACA F5                  <1>       cmc    ; stc
3243 0000AACB C3                  <1>       retn
3244                              <1>
3245                              <1> loc_get_current_drive_not_ready_retn:
3246 0000AACC 5E                  <1>       pop    esi
3247                              <1>       ;mov   eax, 15
3248 0000AACD 66B80F00            <1>       mov    ax, 15 ; Drive not ready
3249 0000AAD1 C3                  <1>       retn
3250                              <1>
3251                              <1> loc_get_current_drive_2:
3252 0000AAD2 31C0                <1>       xor    eax, eax
3253 0000AAD4 88D4                <1>       mov    ah, dl
3254 0000AAD6 56                  <1>       push   esi
3255 0000AAD7 BE00010900          <1>       mov    esi, Logical_DOSDisks
3256 0000AADC 01C6                <1>       add    esi, eax
3257 0000AADE 8A06                <1>       mov    al, [esi+LD_Name]
3258 0000AAE0 3C41                <1>       cmp    al, 'A'
3259 0000AAE2 72E8                <1>       jb     short loc_get_current_drive_not_ready_retn
3260                              <1>
3261 0000AAE4 8A667F              <1>       mov    ah, [esi+LD_CDirLevel]
3262 0000AAE7 08E4                <1>       or     ah, ah
3263 0000AAE9 7506                <1>       jnz    short loc_get_current_drive_3
3264                              <1>
3265                              <1>       ;xor   ah, ah ; mov ah, 0
3266 0000AAEB 8826                <1>       mov    [esi], ah
3267 0000AAED 31C9                <1>       xor    ecx, ecx
3268 0000AAEF EB1C                <1>       jmp    short loc_get_current_drive_4
3269                              <1>
3270                              <1> loc_get_current_drive_3:
3271 0000AAF1 BF[CB880100]        <1>         mov     edi, PATH_Array
3272 0000AAF6 57                  <1>       push   edi
3273 0000AAF7 81C680000000        <1>       add    esi, LD_CurrentDirectory
3274 0000AAFD B920000000          <1>       mov    ecx, 32
3275 0000AB02 F3A5                <1>       rep    movsd
3276 0000AB04 5E                  <1>       pop    esi ; Path Array Address
3277 0000AB05 5F                  <1>       pop    edi ; pushed esi (current dir buffer offset)
3278                              <1>       ;
3279 0000AB06 E841FFFFFF          <1>       call   set_current_directory_string
3280 0000AB0B 89FE                <1>       mov    esi, edi
3281                              <1>
3282                              <1> loc_get_current_drive_4:
3283 0000AB0D 30C0                <1>       xor    al, al
3284 0000AB0F C3                  <1>       retn
3285                              <1>
3286                              <1> change_current_directory:
3287                              <1>       ; 02/03/2021 (TRDOS 386 v2.0.3) ((BugFix))
3288                              <1>       ; 19/02/2016
3289                              <1>       ; 11/02/2016
3290                              <1>       ; 10/02/2016
3291                              <1>       ; 08/02/2016
3292                              <1>       ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
3293                              <1>       ; 18/09/2011 (DIR.ASM, 09/10/2011)
3294                              <1>       ; 04/10/2009
3295                              <1>       ; 2005
3296                              <1>       ; INPUT ->
3297                              <1>       ;      ESI = Directory string
3298                              <1>       ;      ah = CD command (CDh = save current dir string)
3299                              <1>       ; OUTPUT ->
3300                              <1>       ;      EDI = DOS Drive Description Table
3301                              <1>       ;      cf = 1 -> error
3302                              <1>       ;         EAX = Error code
3303                              <1>       ;      cf = 0 -> successful
3304                              <1>       ;         ESI = PATH_Array
3305                              <1>       ;         EAX = Current Directory First Cluster
3306                              <1>       ;
3307                              <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
3308                              <1>
3309 0000AB10 8825[59890100]      <1>       mov    [CD_COMMAND], ah
3310 0000AB16 803E2F              <1>       cmp    byte [esi], '/'
3311 0000AB19 7505                <1>       jne    short loc_ccd_cdir_level
3312 0000AB1B 46                  <1>       inc    esi
3313 0000AB1C 30C0                <1>       xor    al, al
3314 0000AB1E EB05                <1>       jmp    short loc_ccd_parse_path_name
3315                              <1> loc_ccd_cdir_level:
3316 0000AB20 A0[6C810100]        <1>       mov    al, [Current_Dir_Level]
3317                              <1> loc_ccd_parse_path_name:
3318 0000AB25 88C4                <1>       mov    ah, al
3319 0000AB27 BF[CB880100]        <1>       mov    edi, PATH_Array
3320                              <1>
3321                              <1> ; Reset directory levels > cdir level
3322                              <1>       ; is this required !?
3323                              <1>       ;
3324                              <1>       ; Relations:
3325                              <1>       ; MAINPROG.ASM (pass_ccdrv_reset_cdir_FAT_fcluster)
3326                              <1>       ; proc_parse_dir_name,
3327                              <1>       ; proc_change_current_directory (this procedure)
3328                              <1>       ; proc_change_prompt_dir_string
3329                              <1>
3330 0000AB2C 0FB6C8              <1>       movzx  ecx, al
3331 0000AB2F FEC1                <1>       inc    cl
3332 0000AB31 C0E104              <1>       shl    cl, 4
3333 0000AB34 01CF                <1>       add    edi, ecx
3334 0000AB36 B107                <1>       mov    cl, 7
3335 0000AB38 28C1                <1>       sub    cl, al
3336 0000AB3A C0E102              <1>       shl    cl, 2
3337 0000AB3D 89C3                <1>       mov    ebx, eax
3338 0000AB3F 31C0                <1>       xor    eax, eax ; 0
3339 0000AB41 F3AB                <1>       rep    stosd
3340 0000AB43 89D8                <1>       mov    eax, ebx
3341                              <1>
3342 0000AB45 BF[CB880100]        <1>       mov    edi, PATH_Array
3343                              <1>
```

```
3344 0000AB4A 803E20              <1>       cmp   byte [esi], 20h
3345 0000AB4D F5                  <1>       cmc
3346 0000AB4E 7305                <1>       jnc   short pass_ccd_parse_dir_name
3347                              <1>
3348                              <1>             ; ESI = Path name
3349                              <1>             ; AL = CCD_Level
3350 0000AB50 E871010000          <1>       call  parse_dir_name
3351                              <1>             ; AL = CCD_Level
3352                              <1>             ; AH = Last_Dir_Level
3353                              <1>             ; (EDI = PATH_Array)
3354                              <1>
3355                              <1> pass_ccd_parse_dir_name:
3356 0000AB55 9C                  <1>       pushf
3357                              <1>
3358                              <1>       ;mov   [CCD_Level], al
3359                              <1>       ;mov   [Last_Dir_Level], ah
3360 0000AB56 66A3[4F890100]      <1>       mov   [CCD_Level], ax
3361                              <1>
3362 0000AB5C 31DB                <1>       xor   ebx, ebx
3363 0000AB5E 8A3D[6E810100]      <1>       mov   bh, [Current_Drv]
3364 0000AB64 BE00010900          <1>       mov   esi, Logical_DOSDisks
3365 0000AB69 01DE                <1>       add   esi, ebx
3366                              <1>
3367 0000AB6B 9D                  <1>       popf
3368 0000AB6C 720A                <1>       jc    short loc_ccd_bad_path_name_retn
3369                              <1>
3370 0000AB6E 8935[4B890100]      <1>       mov   [CCD_DriveDT], esi
3371                              <1>
3372 0000AB74 3C07                <1>       cmp   al, 7
3373 0000AB76 7209                <1>       jb    short loc_ccd_load_child_dir
3374                              <1>
3375                              <1> loc_ccd_bad_path_name_retn:
3376 0000AB78 87F7                <1>       xchg  esi, edi
3377 0000AB7A B813000000          <1>       mov   eax, 19 ; Bad directory/path name
3378 0000AB7F F9                  <1>       stc
3379                              <1> loc_ccd_retn_p:
3380 0000AB80 C3                  <1>       retn
3381                              <1>
3382                              <1> loc_ccd_load_child_dir:
3383                              <1>       ; AL = CCD_Level
3384 0000AB81 08C0                <1>       or    al, al
3385 0000AB83 7467                <1>       jz    short loc_ccd_load_root_dir
3386                              <1>
3387 0000AB85 6689C1              <1>       mov   cx, ax
3388 0000AB88 C0E004              <1>       shl   al, 4
3389 0000AB8B 0FB6F0              <1>       movzx esi, al
3390 0000AB8E 01FE                <1>       add   esi, edi  ; offset PATH_Array
3391                              <1>
3392 0000AB90 8B460C              <1>       mov   eax, [esi+12]
3393 0000AB93 38E9                <1>       cmp   cl, ch
3394 0000AB95 0F84F9000000        <1>         je      loc_ccd_load_sub_directory
3395 0000AB9B A3[68810100]        <1>       mov   [Current_Dir_FCluster], eax
3396                              <1>
3397                              <1> loc_ccd_load_child_dir_next:
3398 0000ABA0 83C610              <1>       add   esi, 16 ; DOS DirEntry Format FileName Address
3399                              <1>
3400                              <1>       ; Directory attribute : 10h
3401 0000ABA3 B010                <1>       mov   al, 00010000b ; 10h (Attrib AND mask)
3402                              <1>       ;mov   ah, 11001000b ; C8h
3403                              <1>       ; Volume name attribute: 8h
3404 0000ABA5 B408                <1>       mov   ah, 00001000b ; 08h (Attrib NAND, AND --> zero mask)
3405                              <1>
3406                              <1>       ;xor   cx, cx
3407 0000ABA7 31C9                <1>       xor   ecx, ecx ; 02/03/2021
3408 0000ABA9 E8B5010000          <1>       call  locate_current_dir_file
3409 0000ABAE 7353                <1>       jnc   short loc_ccd_set_dir_cluster_ptr
3410                              <1>
3411                              <1>        ; 19/02/2016
3412                              <1>       ;mov   edi, [CCD_DriveDT]
3413 0000ABB0 8A25[4F890100]      <1>       mov   ah, [CCD_Level]
3414 0000ABB6 803D[59890100]CD    <1>       cmp   byte [CD_COMMAND], 0CDh ;'CD' command or another
3415 0000ABBD 7509                <1>       jne   short loc_ccd_load_child_dir_err
3416                              <1>       ; It is better to save recent successful part
3417                              <1>       ; of the (requested) path as current directory.
3418                              <1>       ; (Otherwise the path would be reset to back
3419                              <1>       ; on the next 'CD' command.)
3420 0000ABBF 88E1                <1>       mov   cl, ah
3421 0000ABC1 50                  <1>       push  eax
3422 0000ABC2 E8E3000000          <1>       call  loc_ccd_save_current_dir
3423 0000ABC7 58                  <1>       pop   eax
3424                              <1> loc_ccd_load_child_dir_err:
3425 0000ABC8 3C03                <1>       cmp   al, 3  ; AL = 2 => File not found error
3426 0000ABCA 7202                <1>       jb    short loc_ccd_path_not_found_retn
3427 0000ABCC F9                  <1>       stc
3428 0000ABCD C3                  <1>       retn
3429                              <1>
3430                              <1> loc_ccd_path_not_found_retn:
3431 0000ABCE B003                <1>       mov   al, 3  ; Path not found
3432 0000ABD0 C3                  <1>       retn
3433                              <1>
3434                              <1> loc_ccd_load_FAT_root_dir:
3435 0000ABD1 803D[6D810100]02    <1>       cmp   byte [Current_FATType], 2
3436 0000ABD8 776B                <1>       ja    short loc_ccd_load_FAT32_root_dir
3437                              <1>
3438                              <1>       ;mov   esi, [CCD_DriveDT]
3439                              <1>       ;push  esi
3440 0000ABDA E8B61D0000          <1>       call  load_FAT_root_directory
3441                              <1>       ;pop   edi ; Dos Drv Description Table
3442                              <1>
3443 0000ABDF 89F7                <1>       mov   edi, esi
3444 0000ABE1 BE[CB880100]        <1>       mov   esi, PATH_Array
3445 0000ABE6 7298                <1>       jc    short loc_ccd_retn_p
3446                              <1>
3447 0000ABE8 31C0                <1>       xor   eax, eax
3448 0000ABEA EB78                <1>       jmp   short loc_ccd_set_cdfc
```

```
                                <1>
                                <1> loc_ccd_load_root_dir:
0000ABEC 803D[6D810100]01       <1>         cmp    byte [Current_FATType], 1
0000ABF3 73DC                   <1>         jnb    short loc_ccd_load_FAT_root_dir
                                <1>
                                <1> loc_ccd_load_FS_root_dir:
0000ABF5 E8621E0000             <1>         call   load_FS_root_directory
0000ABFA EB5C                   <1>         jmp    short pass_ccd_load_FAT_sub_directory
                                <1>
                                <1> loc_ccd_load_FS_sub_directory_next:
0000ABFC E85C1E0000             <1>         call   load_FS_sub_directory
0000AC01 EB1F                   <1>         jmp    short pass_ccd_set_dir_cluster_ptr
                                <1>
                                <1> loc_ccd_set_dir_cluster_ptr:
                                <1>         ; EDI = Directory Entry
0000AC03 668B4714               <1>         mov    ax, [edi+20] ; First Cluster High Word
0000AC07 C1E010                 <1>         shl    eax, 16
0000AC0A 668B471A               <1>         mov    ax, [edi+26] ; First Cluster Low Word
                                <1>
0000AC0E 8B35[4B890100]         <1>         mov    esi, [CCD_DriveDT]
0000AC14 803D[6D810100]01       <1>         cmp    byte [Current_FATType], 1
0000AC1B 72DF                   <1>         jb     short loc_ccd_load_FS_sub_directory_next
                                <1>         ;push  esi
0000AC1D E8FE1D0000             <1>         call   load_FAT_sub_directory
                                <1>         ;pop   edi ; Dos Drv Description Table
                                <1>
                                <1> pass_ccd_set_dir_cluster_ptr:
                                <1>         ;mov   edi, esi
0000AC22 BE[CB880100]           <1>         mov    esi, PATH_Array
0000AC27 7264                   <1>         jc     short loc_ccd_retn_c
                                <1>
0000AC29 A1[99880100]           <1>         mov    eax, [DirBuff_Cluster]
                                <1>
0000AC2E FE05[4F890100]         <1>         inc    byte [CCD_Level]
0000AC34 0FB61D[4F890100]       <1>         movzx  ebx, byte [CCD_Level]
0000AC3B C0E304                 <1>         shl    bl, 4 ; * 16 (<= 128)
0000AC3E 01DE                   <1>         add    esi, ebx ; 19/02/2016
0000AC40 89460C                 <1>         mov    [esi+12], eax
0000AC43 EB1F                   <1>         jmp    short loc_ccd_set_cdfc
                                <1>
                                <1> loc_ccd_load_FAT32_root_dir:
0000AC45 BE[CB880100]           <1>         mov    esi, PATH_Array
0000AC4A 8B460C                 <1>         mov    eax, [esi+12]
0000AC4D 8B35[4B890100]         <1>         mov    esi, [CCD_DriveDT]
                                <1>
                                <1> loc_ccd_load_FAT_sub_directory:
                                <1>         ;push  esi
0000AC53 E8C81D0000             <1>         call   load_FAT_sub_directory
                                <1>         ;pop   edi ; Dos Drv Description Table
                                <1>
                                <1> pass_ccd_load_FAT_sub_directory:
                                <1>         ;mov   edi, esi
0000AC58 BE[CB880100]           <1>         mov    esi, PATH_Array
0000AC5D 722E                   <1>         jc     short loc_ccd_retn_c
                                <1>
0000AC5F A1[99880100]           <1>         mov    eax, [DirBuff_Cluster]
                                <1>
                                <1> loc_ccd_set_cdfc:
0000AC64 8A0D[4F890100]         <1>         mov    cl, [CCD_Level]
0000AC6A 880D[6C810100]         <1>         mov    [Current_Dir_Level], cl
0000AC70 A3[68810100]           <1>         mov    [Current_Dir_FCluster], eax
                                <1>
0000AC75 8A2D[50890100]         <1>         mov    ch, [Last_Dir_Level]
0000AC7B 38E9                   <1>         cmp    cl, ch
0000AC7D 0F821DFFFFFF           <1>         jb     loc_ccd_load_child_dir_next
                                <1>
0000AC83 803D[59890100]CD       <1>         cmp    byte [CD_COMMAND], 0CDh ;'CD' command or another
0000AC8A 741E                   <1>         je     short loc_ccd_save_current_dir
                                <1>
                                <1>         ; jne -> don't save, restore (the previous cdir) later !
                                <1>         ; (saving the cdir would prevent previous cdir restoration!)
                                <1>
0000AC8C F8                     <1>         clc
                                <1>
                                <1> loc_ccd_retn_c:
0000AC8D 8B3D[4B890100]         <1>         mov    edi, [CCD_DriveDT]
0000AC93 C3                     <1>         retn
                                <1>
                                <1> loc_ccd_load_sub_directory:
0000AC94 8B35[4B890100]         <1>         mov    esi, [CCD_DriveDT]
0000AC9A 803D[6D810100]01       <1>         cmp    byte [Current_FATType], 1
0000ACA1 73B0                   <1>         jnb    short loc_ccd_load_FAT_sub_directory
0000ACA3 E8B51D0000             <1>         call   load_FS_sub_directory
0000ACA8 EBAE                   <1>         jmp    short pass_ccd_load_FAT_sub_directory
                                <1>
                                <1> loc_ccd_save_current_dir:
                                <1>         ; 02/03/2021 (TRDOS 386 v2.0.3) ((BugFix))
                                <1>         ; ('find_directory_entry' has been fixed to prevent large
                                <1>         ; ECX value > 65535)
                                <1>         ;
0000ACAA BE[CB880100]           <1>         mov    esi, PATH_Array ; 19/02/2016
0000ACAF 8B3D[4B890100]         <1>         mov    edi, [CCD_DriveDT]
0000ACB5 57                     <1>         push   edi
0000ACB6 83C77F                 <1>         add   edi, LD_CDirLevel
0000ACB9 880F                   <1>         mov    [edi], cl
0000ACBB 47                     <1>         inc    edi ; LD_CurrentDirectory
0000ACBC 56                     <1>         push   esi
                                <1>         ;;mov  ecx, 32  ; always < 65536 (in this procedure)
0000ACBD 66B92000               <1>         mov    cx, 32
                                <1>         ; 02/03/2021
                                <1>         ;mov   ecx, 32
0000ACC1 F3A5                   <1>         rep    movsd
                                <1>         ; Current directory has been saved to
                                <1>         ; the DOS drive description table, cdir area !
0000ACC3 5E                     <1>         pop    esi  ; PATH_Array
```

```
3554 0000ACC4 5F                  <1>       pop    edi  ; Dos Drv Description Table
3555                              <1>
3556 0000ACC5 C3                  <1>       retn
3557                              <1>
3558                              <1> parse_dir_name:
3559                              <1>       ; 11/02/2016
3560                              <1>       ; 10/02/2016
3561                              <1>       ; 07/02/2016 (TRDOS 386 = TRDOS v2.0)
3562                              <1>       ; 18/09/2011
3563                              <1>       ; 17/10/2009
3564                              <1>       ; INPUT ->
3565                              <1>       ;      ESI = ASCIIZ Directory String Address
3566                              <1>       ;      AL = Current Directory Level
3567                              <1>       ;      EDI = Destination Adress
3568                              <1>       ;           (8 levels, each one 12+4 byte)
3569                              <1>       ; OUTPUT ->
3570                              <1>       ;      EDI = Dir Entry Formatted Array
3571                              <1>       ;           with zero cluster pointer at the last level
3572                              <1>       ;      AH = Last Dir Level
3573                              <1>       ;      AL = Current Dir Level
3574                              <1>       ;
3575                              <1>       ; (esi, ebx, ecx will be changed)
3576                              <1>
3577                              <1>       ;mov   [PATH_Array_Ptr], edi
3578 0000ACC6 88C4                <1>       mov    ah, al
3579 0000ACC8 66A3[F0890100]      <1>       mov    [PATH_CDLevel], ax
3580                              <1> repeat_ppdn_check_slash:
3581 0000ACCE AC                  <1>       lodsb
3582 0000ACCF 3C2F                <1>       cmp    al, '/'
3583 0000ACD1 74FB                <1>       je     short repeat_ppdn_check_slash
3584 0000ACD3 3C21                <1>       cmp    al, 21h
3585 0000ACD5 7219                <1>       jb     short loc_ppdn_retn
3586 0000ACD7 57                  <1>       push   edi
3587                              <1> loc_ppdn_get_dir_name:
3588 0000ACD8 B90C000000          <1>       mov    ecx, 12
3589 0000ACDD BF[F2890100]        <1>       mov    edi, Dir_File_Name
3590                              <1> repeat_ppdn_get_dir_name:
3591 0000ACE2 AA                  <1>       stosb
3592 0000ACE3 AC                  <1>       lodsb
3593 0000ACE4 3C2F                <1>       cmp    al, '/'
3594 0000ACE6 740A                <1>       je     short loc_check_level_dot_conv_dir_name
3595 0000ACE8 3C20                <1>       cmp    al, 20h
3596 0000ACEA 7605                <1>       jna    short loc_ppdn_end_of_path_scan
3597 0000ACEC E2F4                <1>       loop   repeat_ppdn_get_dir_name
3598 0000ACEE 5F                  <1>       pop    edi
3599 0000ACEF F9                  <1>       stc
3600                              <1> loc_ppdn_retn:
3601 0000ACF0 C3                  <1>       retn
3602                              <1>
3603                              <1> loc_ppdn_end_of_path_scan:
3604 0000ACF1 4E                  <1>       dec    esi
3605                              <1> loc_check_level_dot_conv_dir_name:
3606 0000ACF2 31C0                <1>       xor    eax, eax
3607 0000ACF4 AA                  <1>       stosb
3608 0000ACF5 89F3                <1>       mov    ebx, esi
3609 0000ACF7 BE[F2890100]        <1>       mov    esi, Dir_File_Name
3610 0000ACFC AC                  <1>       lodsb
3611                              <1> repeat_ppdn_name_check_dot:
3612 0000ACFD 3C2E                <1>       cmp    al, '.'
3613 0000ACFF 7509                <1>       jne    short loc_ppdn_convert_sub_dir_name
3614                              <1> repeat_ppdn_name_dot_dot:
3615 0000AD01 AC                  <1>       lodsb
3616 0000AD02 3C2E                <1>       cmp    al, '.'
3617 0000AD04 743E                <1>       je     short loc_ppdn_dot_dot
3618 0000AD06 3C21                <1>       cmp    al, 21h
3619 0000AD08 7226                <1>       jb     short pass_ppdn_convert_sub_dir_name
3620                              <1> loc_ppdn_convert_sub_dir_name:
3621 0000AD0A 8A25[F1890100]      <1>       mov    ah, [PATH_Level]
3622 0000AD10 80FC07              <1>       cmp    ah, 7
3623 0000AD13 731B                <1>       jnb    short pass_ppdn_convert_sub_dir_name
3624 0000AD15 FEC4                <1>       inc    ah
3625 0000AD17 8825[F1890100]      <1>       mov    [PATH_Level], ah
3626 0000AD1D BE[F2890100]        <1>       mov    esi, Dir_File_Name
3627                              <1>       ;mov   edi, [PATH_Array_Ptr]
3628 0000AD22 B010                <1>       mov    al, 16
3629 0000AD24 F6E4                <1>       mul    ah
3630 0000AD26 8B3C24              <1>       mov    edi, [esp]
3631                              <1>       ;push  edi
3632 0000AD29 01C7                <1>       add    edi, eax
3633 0000AD2B E82B030000          <1>       call   convert_file_name
3634                              <1>       ;pop   edi
3635                              <1> pass_ppdn_convert_sub_dir_name:
3636 0000AD30 89DE                <1>       mov    esi, ebx
3637                              <1> repeat_ppdn_check_last_slash:
3638 0000AD32 AC                  <1>       lodsb
3639 0000AD33 3C2F                <1>       cmp    al, '/'
3640 0000AD35 74FB                <1>       je     short repeat_ppdn_check_last_slash
3641 0000AD37 3C21                <1>       cmp    al, 21h
3642 0000AD39 739D                <1>       jnb    short loc_ppdn_get_dir_name
3643                              <1> end_of_parse_dir_name:
3644 0000AD3B 5F                  <1>       pop    edi
3645 0000AD3C F5                  <1>       cmc
3646                              <1>       ;mov   al, [PATH_CDLevel]
3647                              <1>       ;mov   ah, [PATH_Level]
3648 0000AD3D 66A1[F0890100]      <1>       mov    ax, [PATH_CDLevel]
3649 0000AD43 C3                  <1>       retn
3650                              <1>
3651                              <1> loc_ppdn_dot_dot:
3652 0000AD44 AC                  <1>       lodsb
3653 0000AD45 3C21                <1>       cmp    al, 21h
3654 0000AD47 73F2                <1>       jnb    short end_of_parse_dir_name
3655                              <1> loc_ppdn_dot_dot_prev_level:
3656 0000AD49 66A1[F0890100]      <1>       mov    ax, [PATH_CDLevel]
3657 0000AD4F 80EC01              <1>       sub    ah, 1
3658 0000AD52 80D400              <1>       adc    ah, 0
```

```
3659 0000AD55 38E0                <1>       cmp    al, ah
3660 0000AD57 7602                <1>       jna    short pass_ppdn_set_al_to_ah
3661 0000AD59 88E0                <1>       mov    al, ah
3662                              <1> pass_ppdn_set_al_to_ah:
3663 0000AD5B 66A3[F0890100]      <1>       mov    [PATH_CDLevel], ax
3664 0000AD61 EBCD                <1>       jmp    short pass_ppdn_convert_sub_dir_name
3665                              <1>
3666                              <1> locate_current_dir_file:
3667                              <1>       ; 20/11/2017
3668                              <1>       ; 14/02/2016
3669                              <1>       ; 13/02/2016
3670                              <1>       ; 10/02/2016
3671                              <1>       ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
3672                              <1>       ; 14/08/2010
3673                              <1>       ; 19/09/2009
3674                              <1>         ; 2005
3675                              <1>       ; INPUT ->
3676                              <1>       ;      ESI = DOS DirEntry Format FileName Address
3677                              <1>       ;      AL = Attributes Mask
3678                              <1>       ;      (<AL AND EntryAttrib> must be equal to AL)
3679                              <1>       ;      AH = Negative Attributes Mask (If AH>0)
3680                              <1>       ;      (<AH AND EntryAttrib> must be ZERO)
3681                              <1>       ;      CH > 0 Find First Free Dir Entry or Deleted Entry
3682                              <1>       ;      CL = 0 -> Return the First Free Dir Entry
3683                              <1>       ;      CL = E5h -> Return the 1st deleted entry
3684                              <1>       ;      CL = FFh -> Return the 1st deleted or free entry
3685                              <1>       ;      CL > 0 and CL <> E5h and CL <> FFh -> Return the first
3686                              <1>       ;           proper entry (which fits with Atributes Masks)
3687                              <1>       ;      CX = 0 Find Valid File/Directory/VolumeName
3688                              <1>       ;      ? = Any One Char
3689                              <1>       ;      * = Every Chars
3690                              <1>       ; OUTPUT ->
3691                              <1>       ;      EDI = Directory Entry Address (in Directory Buffer)
3692                              <1>       ;      ESI = DOS DirEntry Format FileName Address
3693                              <1>       ;      CF = 0 -> No Error, Proper Entry,
3694                              <1>       ;      DL = Attributes
3695                              <1>       ;      DH = Previous Entry Attr (LongName Check)
3696                              <1>       ;      AL > 0 -> Ambiguous filename wildcard "?" used
3697                              <1>       ;      AH > 0 -> Ambiguous filename wildcard "*" used
3698                              <1>       ;      AX = 0 -> Filename full fits with directory entry
3699                              <1>       ;      CH = The 1st Name Char of Current Dir Entry
3700                              <1>       ;      CF = 1 -> Proper entry not found, Error Code in EAX/AL
3701                              <1>       ;      CL = 0 and CH = 0 -> Free Entry (End Of Dir)
3702                              <1>       ;      CL = 0 and CH = E5h -> Deleted Entry fits with filters
3703                              <1>       ;      CL > 0 -> Entry not found, CH invalid
3704                              <1>       ;      CF = 0 ->
3705                              <1>       ;      EBX = Current Directory Entry Index/Number (BX)
3706                              <1>
3707                              <1>       ;mov   word [DirBuff_EntryCounter], 0 ; Zero Based
3708                              <1>
3709 0000AD63 8935[53890100]      <1>       mov    [CDLF_FNAddress], esi
3710 0000AD69 66A3[51890100]      <1>       mov    [CDLF_AttributesMask], ax
3711 0000AD6F 66890D[57890100]    <1>       mov    [CDLF_DEType], cx
3712                              <1>
3713 0000AD76 31DB                <1>       xor    ebx, ebx
3714 0000AD78 881D[68890100]      <1>       mov    [PreviousAttr], bl ; 0  ; 13/02/2016
3715                              <1>
3716 0000AD7E 8A3D[6E810100]      <1>       mov    bh, [Current_Drv]
3717 0000AD84 381D[94880100]      <1>       cmp    byte [DirBuff_ValidData], bl ; 0
3718 0000AD8A 761D                <1>       jna    short loc_lcdf_reload_current_dir2
3719 0000AD8C 8A1D[92880100]      <1>       mov   bl, [DirBuff_DRV]
3720 0000AD92 80EB41              <1>       sub    bl, 'A'
3721 0000AD95 38DF                <1>       cmp    bh, bl
3722 0000AD97 750E                <1>       jne    short loc_lcdf_reload_current_dir1
3723 0000AD99 8B15[99880100]      <1>       mov    edx, [DirBuff_Cluster]
3724 0000AD9F 3B15[68810100]      <1>       cmp    edx, [Current_Dir_FCluster]
3725 0000ADA5 7412                <1>       je     short loc_cdir_locatefile_search
3726                              <1>
3727                              <1> loc_lcdf_reload_current_dir1:
3728 0000ADA7 30DB                <1>       xor    bl, bl
3729                              <1> loc_lcdf_reload_current_dir2:
3730 0000ADA9 89DE                <1>       mov    esi, ebx
3731 0000ADAB 81C600010900        <1>       add   esi, Logical_DOSDisks
3732 0000ADB1 E874000000          <1>       call   reload_current_directory
3733 0000ADB6 735D                <1>       jnc    short loc_locatefile_search_again
3734 0000ADB8 C3                  <1>       retn
3735                              <1>
3736                              <1> loc_cdir_locatefile_search:
3737 0000ADB9 31DB                <1>       xor    ebx, ebx
3738 0000ADBB 55                  <1>       push   ebp ; 20/11/2017
3739 0000ADBC E8A6000000          <1>       call   find_directory_entry
3740 0000ADC1 5D                  <1>       pop    ebp ; 20/11/2017
3741 0000ADC2 7349                <1>       jnc    short loc_cdir_locate_file_retn
3742                              <1>
3743                              <1> loc_locatefile_check_stc_reason:
3744 0000ADC4 08ED                <1>       or     ch, ch
3745 0000ADC6 7444                <1>       jz     short loc_cdir_locate_file_stc_retn
3746                              <1>
3747                              <1> loc_locatefile_check_next_entryblock:
3748 0000ADC8 8A3D[6E810100]      <1>       mov    bh, [Current_Drv]
3749 0000ADCE 28DB                <1>       sub    bl, bl
3750 0000ADD0 0FB7F3              <1>       movzx  esi, bx
3751 0000ADD3 81C600010900        <1>       add   esi, Logical_DOSDisks
3752                              <1>
3753 0000ADD9 803D[6C810100]00    <1>       cmp    byte [Current_Dir_Level], 0
3754 0000ADE0 760A                <1>       jna    short loc_locatefile_check_FAT_type
3755                              <1>
3756 0000ADE2 803D[6D810100]01    <1>       cmp    byte [Current_FATType], 1
3757 0000ADE9 730A                <1>       jnb    short loc_locatefile_load_subdir_cluster
3758 0000ADEB C3                  <1>       retn
3759                              <1>
3760                              <1> loc_locatefile_check_FAT_type:
3761 0000ADEC 803D[6D810100]03    <1>       cmp    byte [Current_FATType], 3
3762 0000ADF3 7218                <1>       jb     short loc_cdir_locate_file_retn
3763                              <1>
```

```
3764                                <1> loc_locatefile_load_subdir_cluster:
3765 0000ADF5 A1[99880100]         <1>         mov    eax, [DirBuff_Cluster]
3766 0000ADFA E83B1A0000           <1>         call   get_next_cluster
3767 0000ADFF 730D                 <1>         jnc    short loc_locatefile_next_cluster
3768 0000AE01 09C0                 <1>         or     eax, eax
3769 0000AE03 7507                 <1>         jnz    short loc_locatefile_drive_not_ready_read_err
3770 0000AE05 F9                   <1>         stc
3771                                <1> loc_locatefile_file_notfound:
3772 0000AE06 B802000000           <1>         mov    eax, 2 ; File/Directory/VolName not found
3773 0000AE0B C3                   <1>         retn
3774                                <1>
3775                                <1> loc_locatefile_drive_not_ready_read_err:
3776                                <1>         ;mov   eax, 17 ;Drive not ready or read error
3777                                <1> loc_cdir_locate_file_stc_retn:
3778 0000AE0C F5                   <1>         cmc ;stc
3779                                <1> loc_cdir_locate_file_retn:
3780 0000AE0D C3                   <1>         retn
3781                                <1>
3782                                <1> loc_locatefile_next_cluster:
3783 0000AE0E E80D1C0000           <1>         call   load_FAT_sub_directory
3784                                <1>         ;jc    short loc_locatefile_drive_not_ready_read_err
3785 0000AE13 72F8                 <1>         jc     short loc_cdir_locate_file_retn
3786                                <1>
3787                                <1> loc_locatefile_search_again:
3788 0000AE15 8B35[53890100]       <1>         mov    esi, [CDLF_FNAddress]
3789 0000AE1B 66A1[51890100]       <1>         mov    ax, [CDLF_AttributesMask]
3790 0000AE21 668B0D[57890100]     <1>         mov    cx, [CDLF_DEType]
3791 0000AE28 EB8F                 <1>         jmp    short loc_cdir_locatefile_search
3792                                <1>
3793                                <1> reload_current_directory:
3794                                <1>         ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
3795                                <1>         ; 13/06/2010
3796                                <1>         ; 22/09/2009
3797                                <1>         ;
3798                                <1>         ; INPUT ->
3799                                <1>         ;      ESI = Dos drive description table address
3800                                <1>
3801                                <1>         ;mov   al, [esi+LD_FATType]
3802 0000AE2A A0[6D810100]         <1>         mov    al, [Current_FATType]
3803 0000AE2F 3C02                 <1>         cmp    al, 2
3804 0000AE31 7729                 <1>         ja     short loc_reload_FAT_sub_directory
3805 0000AE33 8A25[6C810100]       <1>         mov    ah, [Current_Dir_Level]
3806 0000AE39 08C0                 <1>         or     al, al
3807 0000AE3B 740A                 <1>         jz     short loc_reload_FS_directory
3808 0000AE3D 08E4                 <1>         or     ah, ah
3809 0000AE3F 751B                 <1>         jnz    short loc_reload_FAT_sub_directory
3810                                <1> loc_reload_FAT_12_16_root_directory:
3811 0000AE41 E84F1B0000           <1>         call   load_FAT_root_directory
3812 0000AE46 C3                   <1>         retn
3813                                <1> loc_reload_FS_directory:
3814 0000AE47 20E4                 <1>         and    ah, ah
3815 0000AE49 7506                 <1>         jnz    short loc_reload_FS_sub_directory
3816                                <1> loc_reload_FS_root_directory:
3817 0000AE4B E80C1C0000           <1>         call   load_FS_root_directory
3818 0000AE50 C3                   <1>         retn
3819                                <1> loc_reload_FS_sub_directory:
3820 0000AE51 A1[68810100]         <1>         mov    eax, [Current_Dir_FCluster]
3821 0000AE56 E8021C0000           <1>         call   load_FS_sub_directory
3822 0000AE5B C3                   <1>         retn
3823                                <1> loc_reload_FAT_sub_directory:
3824 0000AE5C A1[68810100]         <1>         mov    eax, [Current_Dir_FCluster]
3825 0000AE61 E8BA1B0000           <1>         call   load_FAT_sub_directory
3826 0000AE66 C3                   <1>         retn
3827                                <1>
3828                                <1> find_directory_entry:
3829                                <1>         ; 02/03/2021 (TRDOS 386 v2.0.3) ((BugFix))
3830                                <1>         ; 14/02/2016
3831                                <1>         ; 13/02/2016
3832                                <1>         ; 10/02/2016
3833                                <1>         ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
3834                                <1>         ; 14/08/2010 (DIR.ASM, "proc_find_direntry")
3835                                <1>         ; 19/09/2009
3836                                <1>         ; 2005
3837                                <1>         ; INPUT ->
3838                                <1>         ;      ESI = Sub Dir or File Name Address
3839                                <1>         ;      AL = Attributes Mask
3840                                <1>         ;      (<AL AND EntryAttrib> must be equal to AL)
3841                                <1>         ;      AH = Negative Attributes Mask (If AH>0)
3842                                <1>         ;      (<AH AND EntryAttrib> must be ZERO)
3843                                <1>         ;      CH > 0 Find First Free Dir Entry or Deleted Entry
3844                                <1>         ;      CL = 0 -> Return the First Free Dir Entry
3845                                <1>         ;      CL = E5h -> Return the 1st deleted entry
3846                                <1>         ;      CL = FFh -> Return the 1st deleted or free entry
3847                                <1>         ;      CL > 0 and CL <> E5h and CL <> FFh -> Return the first
3848                                <1>         ;            proper entry (which fits with Atributes Masks)
3849                                <1>         ;      CX = 0 -> Find Valid File/Directory/VolumeName
3850                                <1>         ;      ? = Any One Char
3851                                <1>         ;      * = Every Chars
3852                                <1>         ;      EBX = Current Dir Entry (BX)
3853                                <1>         ;
3854                                <1>         ; OUTPUT ->
3855                                <1>         ;      EDI = Directory Entry Address (in DirectoryBuffer)
3856                                <1>         ;      ESI = Sub Dir or File Name Address
3857                                <1>         ;      CF = 0 -> No Error, Proper Entry,
3858                                <1>         ;      DL = Attributes
3859                                <1>         ;      DH = Previous Entry Attr (LongName Check)
3860                                <1>         ;      AL > 0 -> Ambiguous filename wildcard "?" used
3861                                <1>         ;      AH > 0 -> Ambiguous filename wildcard "*" used
3862                                <1>         ;      AX = 0 -> Filename full fits with directory entry
3863                                <1>         ;      EBX = CurrentDirEntry (BX)
3864                                <1>         ;      CH = The 1st Name Char of Current Dir Entry
3865                                <1>         ;      CF = 1 -> Proper entry not found, Error Code in AX/AL
3866                                <1>         ;      CL = 0 and CH = 0 -> Free Entry (End Of Dir)
3867                                <1>         ;      CL = 0 and CH = E5h -> Deleted Entry fits with filters
3868                                <1>         ;      CL > 0 -> Entry not found, CH invalid
```

```
3869                                    <1>      ;
3870                                    <1>      ; (EAX, EBX, ECX, EDX, EDI, EBP will be changed)
3871                                    <1>
3872 0000AE67 663B1D[97880100]         <1>      cmp    bx, [DirBuff_LastEntry]
3873 0000AE6E 0F8739010000             <1>      ja     loc_ffde_stc_retn_255
3874                                    <1>
3875                                    <1>      ;mov  [DirBuff_CurrentEntry], bx
3876                                    <1>
3877 0000AE74 BF00000800               <1>      mov    edi, Directory_Buffer
3878 0000AE79 66A3[64890100]           <1>      mov    [FDE_AttrMask], ax
3879                                    <1>
3880 0000AE7F 29C0                     <1>      sub    eax, eax
3881                                    <1>
3882                                    <1>      ;;mov  [PreviousAttr], al ; 0 ;; 13/02/2016
3883 0000AE81 66A3[66890100]           <1>      mov    [AmbiguousFileName], ax ; 0
3884                                    <1>
3885 0000AE87 6689D8                   <1>      mov    ax, bx
3886 0000AE8A 66C1E005                 <1>      shl    ax, 5 ; ; * 32 ; Directory entry size
3887 0000AE8E 01C7                     <1>      add     edi, eax
3888                                    <1>
3889 0000AE90 08ED                     <1>      or     ch, ch
3890 0000AE92 0F852D010000             <1>      jnz    loc_find_free_deleted_entry_0
3891                                    <1>
3892 0000AE98 08C9                     <1>      or    cl, cl
3893 0000AE9A 0F850D010000             <1>      jnz    loc_ffde_stc_retn_255
3894                                    <1>
3895                                    <1> check_find_dir_entry:
3896 0000AEA0 66A1[64890100]           <1>      mov    ax, [FDE_AttrMask]
3897 0000AEA6 8A2F                     <1>      mov    ch, [edi]
3898 0000AEA8 80FD00                   <1>      cmp    ch, 0 ; Is it never used entry?
3899 0000AEAB 0F8600010000             <1>      jna    loc_find_direntry_stc_retn
3900 0000AEB1 56                       <1>      push   esi
3901 0000AEB2 8A570B                   <1>      mov    dl, [edi+0Bh] ; File attributes
3902 0000AEB5 80FDE5                   <1>      cmp    ch, 0E5h ; Is it a deleted file?
3903 0000AEB8 746D                     <1>      je     short loc_find_dir_next_entry_prevdeleted
3904                                    <1>
3905 0000AEBA 80FA0F                   <1>      cmp    dl, 0Fh ; longname sub component check
3906 0000AEBD 7505                     <1>      jne    short loc_check_attributes_mask
3907 0000AEBF E8EE010000               <1>      call   save_longname_sub_component
3908                                    <1>
3909                                    <1> loc_check_attributes_mask:
3910 0000AEC4 88C6                     <1>      mov    dh, al
3911 0000AEC6 20D6                     <1>      and    dh, dl
3912 0000AEC8 38F0                     <1>      cmp    al, dh
3913 0000AECA 0F85BA000000             <1>      jne   loc_find_dir_next_entry
3914 0000AED0 20D4                     <1>      and    ah, dl
3915 0000AED2 0F85B2000000             <1>      jnz   loc_find_dir_next_entry
3916 0000AED8 80FA0F                   <1>      cmp    dl, 0Fh
3917 0000AEDB 751A                     <1>      jne    short pass_direntry_attr_check
3918                                    <1>
3919 0000AEDD 3C0F                     <1>      cmp    al, 0Fh ; AL = 0Fh -> find long name
3920 0000AEDF 0F85A5000000             <1>      jne   loc_find_dir_next_entry
3921                                    <1>
3922 0000AEE5 5E                       <1>      pop    esi
3923 0000AEE6 6631C0                   <1>      xor    ax, ax
3924 0000AEE9 8A35[68890100]           <1>      mov    dh, [PreviousAttr]
3925 0000AEEF 66891D[95880100]         <1>      mov    [DirBuff_CurrentEntry], bx
3926 0000AEF6 C3                       <1>      retn
3927                                    <1>
3928                                    <1> pass_direntry_attr_check:
3929 0000AEF7 89FD                     <1>      mov    ebp, edi ; 14/02/2016
3930 0000AEF9 B908000000               <1>      mov    ecx, 8
3931                                    <1> loc_lodsb_find_dir:
3932 0000AEFE AC                       <1>      lodsb
3933 0000AEFF 3C2A                     <1>      cmp    al, '*'
3934 0000AF01 7508                     <1>      jne    short pass_fde_ambiguous1_check
3935 0000AF03 FE05[67890100]           <1>      inc   byte [AmbiguousFileName+1]
3936 0000AF09 EB28                     <1>      jmp    short loc_check_direntry_extension
3937                                    <1>
3938                                    <1> pass_fde_ambiguous1_check:
3939 0000AF0B 3C3F                     <1>      cmp    al, '?'
3940 0000AF0D 750D                     <1>      jne    short pass_fde_ambiguous2_check
3941 0000AF0F FE05[66890100]           <1>      inc    byte [AmbiguousFileName]
3942 0000AF15 803F20                   <1>      cmp    byte [edi], 20h
3943 0000AF18 764E                     <1>      jna    short loc_find_dir_next_entry_ebp
3944 0000AF1A EB14                     <1>      jmp    short loc_scasb_find_dir_inc_di
3945                                    <1>
3946                                    <1> pass_fde_ambiguous2_check:
3947 0000AF1C 3C20                     <1>      cmp    al, 20h
3948 0000AF1E 750C                     <1>      jne    short loc_scasb_find_dir
3949 0000AF20 803F20                   <1>      cmp    byte [edi], 20h
3950 0000AF23 7543                     <1>      jne    short loc_find_dir_next_entry_ebp
3951 0000AF25 EB0C                     <1>      jmp    short loc_check_direntry_extension
3952                                    <1>
3953                                    <1> loc_find_dir_next_entry_prevdeleted:
3954 0000AF27 80CA80                   <1>      or     dl, 80h  ; Bit 7 -> deleted entry sign
3955 0000AF2A EB5E                     <1>      jmp    short loc_find_dir_next_entry
3956                                    <1>
3957                                    <1> loc_scasb_find_dir:
3958 0000AF2C 3A07                     <1>      cmp    al, [edi]
3959 0000AF2E 7538                     <1>      jne    short loc_find_dir_next_entry_ebp
3960                                    <1> loc_scasb_find_dir_inc_di:
3961 0000AF30 47                       <1>      inc    edi
3962 0000AF31 E2CB                     <1>      loop   loc_lodsb_find_dir
3963                                    <1>
3964                                    <1> loc_check_direntry_extension:
3965 0000AF33 BE08000000               <1>      mov    esi, 8
3966 0000AF38 89F7                     <1>      mov    edi, esi ; 8
3967 0000AF3A 033424                   <1>      add    esi, [esp] ; Sub Dir or File Name Address
3968 0000AF3D 01EF                     <1>      add    edi, ebp
3969 0000AF3F B103                     <1>      mov    cl, 3
3970                                    <1> loc_lodsb_find_dir_ext:
3971 0000AF41 AC                       <1>      lodsb
3972 0000AF42 3C2A                     <1>      cmp    al, '*'
3973 0000AF44 7508                     <1>      jne    short pass_fde_ambiguous3_check
```

```
3974 0000AF46 FE05[67890100]      <1>       inc    byte [AmbiguousFileName+1]
3975 0000AF4C EB1E                <1>       jmp    short loc_find_dir_proper_direntry
3976                              <1>
3977                              <1> pass_fde_ambiguous3_check:
3978 0000AF4E 3C3F                <1>       cmp    al, '?'
3979 0000AF50 750D                <1>       jne    short pass_fde_ambiguous4_check
3980 0000AF52 FE05[66890100]      <1>       inc    byte [AmbiguousFileName]
3981 0000AF58 803F20              <1>       cmp    byte [edi], 20h
3982 0000AF5B 760B                <1>       jna    short loc_find_dir_next_entry_ebp
3983 0000AF5D EB49                <1>       jmp    short loc_scasb_find_dir_ext_inc_di
3984                              <1>
3985                              <1> pass_fde_ambiguous4_check:
3986 0000AF5F 3C20                <1>       cmp    al, 20h
3987 0000AF61 7541                <1>       jne    short loc_scasb_find_dir_ext
3988 0000AF63 803F20              <1>       cmp    byte [edi], 20h
3989 0000AF66 7404                <1>       je     short loc_find_dir_proper_direntry
3990                              <1>
3991                              <1> loc_find_dir_next_entry_ebp:
3992 0000AF68 89EF                <1>       mov    edi, ebp ; 14/02/2016
3993 0000AF6A EB1E                <1>       jmp    short loc_find_dir_next_entry
3994                              <1>
3995                              <1> loc_find_dir_proper_direntry:
3996 0000AF6C 30C9                <1>       xor    cl, cl
3997                              <1> loc_find_dir_proper_direntry_1:
3998 0000AF6E 5E                  <1>       pop    esi
3999 0000AF6F 89EF                <1>       mov   edi, ebp
4000 0000AF71 8A2F                <1>       mov    ch, [edi]
4001 0000AF73 8A570B              <1>       mov   dl, [edi+0Bh] ; Dir entry attributes
4002 0000AF76 66A1[66890100]      <1>       mov    ax, [AmbiguousFileName]
4003                              <1> loc_find_dir_proper_direntry_2:
4004 0000AF7C 8A35[68890100]      <1>       mov     dh, [PreviousAttr]
4005 0000AF82 66891D[95880100]    <1>       mov    [DirBuff_CurrentEntry], bx
4006 0000AF89 C3                  <1>       retn
4007                              <1>
4008                              <1> loc_find_dir_next_entry:
4009 0000AF8A 8815[68890100]      <1>       mov   byte [PreviousAttr], dl ; LongName check
4010                              <1> loc_find_dir_next_entry_1:
4011 0000AF90 5E                  <1>       pop    esi
4012 0000AF91 83C720              <1>       add    edi, 32
4013                              <1>       ;inc   word [DirBuff_EntryCounter]
4014 0000AF94 6643                <1>       inc    bx
4015 0000AF96 663B1D[97880100]    <1>       cmp    bx, [DirBuff_LastEntry]
4016 0000AF9D 770E                <1>       ja     short loc_ffde_stc_retn_255
4017 0000AF9F E9FCFEFFFF          <1>       jmp   check_find_dir_entry
4018                              <1>
4019                              <1> loc_scasb_find_dir_ext:
4020 0000AFA4 3A07                <1>       cmp    al, [edi]
4021 0000AFA6 75C0                <1>       jne    short loc_find_dir_next_entry_ebp
4022                              <1> loc_scasb_find_dir_ext_inc_di:
4023 0000AFA8 47                  <1>       inc    edi
4024 0000AFA9 E296                <1>       loop  loc_lodsb_find_dir_ext
4025 0000AFAB EBC1                <1>       jmp    short loc_find_dir_proper_direntry_1
4026                              <1>
4027                              <1> loc_ffde_stc_retn_255:
4028                              <1>       ; 02/03/2021 (TRDOS 386 v2.0.3) ((BugFix))
4029                              <1>       ; (ECX must not be > 65535)
4030                              <1>       ; ((because 'loc_ccd_save_current_dir'
4031                              <1>       ;  sets CX to 32 for 'rep movsd'))
4032 0000AFAD 66B9FFFF            <1>       mov    cx, 0FFFFh
4033                              <1>       ;xor   ecx, ecx
4034                              <1>       ;dec   ecx ; 0FFFFFFFFh
4035                              <1>       ;xor   eax, eax
4036                              <1> loc_find_direntry_stc_retn:
4037                              <1> loc_check_ffde_retn_1:
4038                              <1>       ;mov   ax, 2
4039 0000AFB1 B802000000          <1>       mov    eax, 2 ; File Not Found
4040 0000AFB6 8A35[68890100]      <1>       mov    dh, [PreviousAttr]
4041 0000AFBC 66891D[95880100]    <1>       mov    [DirBuff_CurrentEntry], bx
4042 0000AFC3 F9                  <1>       stc
4043 0000AFC4 C3                  <1>       retn
4044                              <1>
4045                              <1> loc_find_free_deleted_entry_0:
4046 0000AFC5 66A1[64890100]      <1>       mov    ax, [FDE_AttrMask]
4047 0000AFCB 8A2F                <1>       mov    ch, [edi]
4048 0000AFCD 8A570B              <1>       mov    dl, [edi+0Bh] ; File attributes
4049 0000AFD0 08C9                <1>       or     cl, cl
4050 0000AFD2 7407                <1>       jz     short loc_check_ffde_0_repeat
4051                              <1>       ;cmp   cl, 0E5h
4052                              <1>       ;je    short pass_loc_check_ffde_0_err
4053 0000AFD4 80F9FF              <1>       cmp    cl, 0FFh
4054 0000AFD7 7432                <1>       je     short loc_find_free_deleted_entry_1
4055 0000AFD9 EB4D                <1>       jmp    short pass_loc_check_ffde_0_err
4056                              <1>
4057                              <1> loc_check_ffde_0_repeat:
4058 0000AFDB 08ED                <1>       or     ch, ch
4059 0000AFDD 7511                <1>       jnz    short loc_check_ffde_0_next
4060                              <1>
4061                              <1> loc_check_ffde_retn_2:
4062 0000AFDF 6629C0              <1>       sub    ax, ax
4063 0000AFE2 8A35[68890100]      <1>       mov    dh, [PreviousAttr]
4064 0000AFE8 66891D[95880100]    <1>       mov    [DirBuff_CurrentEntry], bx
4065 0000AFEF C3                  <1>       retn
4066                              <1>
4067                              <1> loc_check_ffde_0_next:
4068 0000AFF0 6643                <1>       inc    bx
4069 0000AFF2 83C720              <1>       add    edi, 32
4070                              <1>       ;inc   word [DirBuff_EntryCounter]
4071                              <1>
4072 0000AFF5 663B1D[97880100]    <1>       cmp    bx, [DirBuff_LastEntry]
4073 0000AFFC 77AF                <1>       ja     short loc_ffde_stc_retn_255
4074 0000AFFE 8815[68890100]      <1>       mov    [PreviousAttr], dl
4075 0000B004 8A2F                <1>       mov    ch, [edi]
4076 0000B006 8A570B              <1>       mov    dl, [edi+0Bh] ; file attributes
4077 0000B009 EBD0                <1>       jmp    short loc_check_ffde_0_repeat
4078                              <1>
```

```
                                <1> loc_find_free_deleted_entry_1:
0000B00B 28D2                   <1>       sub    dl, dl
                                <1> loc_find_free_deleted_entry_2:
0000B00D 20ED                   <1>       and    ch, ch
0000B00F 74CE                   <1>       jz     short loc_check_ffde_retn_2
0000B011 80FDE5                 <1>       cmp    ch, 0E5h
0000B014 74C9                   <1>       je     short loc_check_ffde_retn_2
0000B016 6643                   <1>       inc    bx
0000B018 83C720                 <1>       add    edi, 32
0000B01B 663B1D[97880100]       <1>       cmp    bx, [DirBuff_LastEntry]
0000B022 7789                   <1>       ja     short loc_ffde_stc_retn_255
0000B024 8A2F                   <1>       mov    ch, [edi]
0000B026 EBE5                   <1>       jmp    short loc_find_free_deleted_entry_2
                                <1>
                                <1> pass_loc_check_ffde_0_err:
0000B028 38CD                   <1>       cmp    ch, cl
0000B02A 741F                   <1>       je     short loc_check_ffde_attrib
                                <1>
0000B02C 6643                   <1>       inc    bx
0000B02E 83C720                 <1>       add    edi, 32
0000B031 663B1D[97880100]       <1>       cmp    bx, [DirBuff_LastEntry]
0000B038 0F876FFFFFFF           <1>       ja    loc_ffde_stc_retn_255
0000B03E 8815[68890100]         <1>       mov    [PreviousAttr], dl
0000B044 8A2F                   <1>       mov    ch, [edi]
0000B046 8A570B                 <1>       mov    dl, [edi+0Bh]
0000B049 EBDD                   <1>       jmp    short pass_loc_check_ffde_0_err
                                <1>
                                <1> loc_check_ffde_attrib:
0000B04B 88C6                   <1>       mov    dh, al
0000B04D 20D6                   <1>       and    dh, dl
0000B04F 38F0                   <1>       cmp    al, dh
0000B051 759D                   <1>       jne    short loc_check_ffde_0_next
0000B053 20D4                   <1>       and    ah, dl
0000B055 7599                   <1>       jnz    short loc_check_ffde_0_next
0000B057 30C9                   <1>       xor    cl, cl
0000B059 EB84                   <1>       jmp   loc_check_ffde_retn_2
                                <1>
                                <1> convert_file_name:
                                <1>       ; 06/03/2016
                                <1>       ; 11/02/2016
                                <1>       ; 07/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 06/10/2009
                                <1>       ; 2005
                                <1>       ;
                                <1>       ; INPUT  ->
                                <1>       ;      ESI = Dot File Name Location
                                <1>       ;      EDI = Dir Entry Format File Name Location
                                <1>       ; OUTPUT ->
                                <1>       ;      EDI = Dir Entry Format File Name Location
                                <1>       ;      ESI = Dot File Name Location (capitalized)
                                <1>       ;
                                <1>       ; (ECX, AL will be changed)
                                <1>
0000B05B 56                     <1>       push   esi
0000B05C 57                     <1>       push   edi
                                <1>
0000B05D B90B000000             <1>       mov    ecx, 11
0000B062 B020                   <1>       mov    al, 20h
0000B064 F3AA                   <1>       rep    stosb
                                <1>
0000B066 8B3C24                 <1>       mov    edi, [esp]
                                <1>
0000B069 B10C                   <1>       mov    cl, 12 ; file name length (max.)
                                <1>       ; 06/03/2016
                                <1>       ; Directory entry name limit (11 bytes) check for
                                <1>       ; 'rename_directory_entry' procedure.
                                <1>       ; (EDI points to Directory Entry)
                                <1>       ; (If the file name would not contain a dot
                                <1>       ; and file name length would be 12, this would cause to
                                <1>       ; overwrite the attributes byte of the directory entry.)
                                <1>       ;
0000B06B B50B                   <1>       mov    ch, 11 ; directory entry's name length
                                <1> loc_check_first_dot:
0000B06D 8A06                   <1>       mov    al, [esi]
0000B06F 3C2E                   <1>       cmp    al, 2Eh
0000B071 750C                   <1>       jne    short pass_check_first_dot
0000B073 8807                   <1>       mov    [edi], al
0000B075 47                     <1>       inc    edi
0000B076 46                     <1>       inc    esi
0000B077 FEC9                   <1>       dec    cl
0000B079 75F2                   <1>       jnz    short loc_check_first_dot
                                <1>       ;;(ecx <= 12)
                                <1>       ;;loop loc_check_first_dot
0000B07B EB30                   <1>       jmp    short stop_convert_file
                                <1>
                                <1> loc_get_fchar:
0000B07D 8A06                   <1>       mov    al, [esi]
                                <1> pass_check_first_dot:
0000B07F 3C61                   <1>       cmp    al, 61h ; 'a'
0000B081 7208                   <1>       jb     short pass_name_capitalize
0000B083 3C7A                   <1>       cmp    al, 7Ah ; 'z'
0000B085 7704                   <1>       ja     short pass_name_capitalize
0000B087 24DF                   <1>       and    al, 0DFh
0000B089 8806                   <1>       mov    [esi], al
                                <1> pass_name_capitalize:
0000B08B 3C21                   <1>       cmp    al, 21h
0000B08D 721E                   <1>       jb     short stop_convert_file
0000B08F 3C2E                   <1>       cmp    al, 2Eh ; '.'
0000B091 750C                   <1>       jne    short pass_dot_space
                                <1> add_dot_space:
0000B093 80F904                 <1>       cmp    cl, 4
0000B096 760E                   <1>       jna    short inc_and_loop
0000B098 47                     <1>       inc    edi
0000B099 FECD                   <1>       dec    ch ; 06/03/2016
0000B09B FEC9                   <1>       dec    cl
```

```
4184 0000B09D EBF4              <1>       jmp    short add_dot_space
4185                            <1>
4186                            <1>       ;mov   al, 4
4187                            <1>       ;cmp   cl, al
4188                            <1>       ;jna   short inc_and_loop
4189                            <1>       ;sub   cl, al
4190                            <1>       ;add   edi, ecx
4191                            <1>       ;mov   cl, al
4192                            <1>       ;jmp   short inc_and_loop
4193                            <1>
4194                            <1> pass_dot_space:
4195 0000B09F 8807              <1>       mov    [edi], al
4196                            <1> loc_after_double_dot:
4197                            <1>       ; 06/03/2016
4198 0000B0A1 FECD              <1>       dec    ch ; count down for 11 bytes dir entry limit
4199 0000B0A3 740A              <1>       jz     short stop_convert_file_x
4200 0000B0A5 47                <1>       inc    edi
4201                            <1> inc_and_loop:
4202 0000B0A6 FEC9              <1>       dec    cl ; count down for 12 bytes filename limit
4203 0000B0A8 7403              <1>       jz     short stop_convert_file
4204 0000B0AA 46                <1>       inc    esi
4205                            <1>       ;;(ecx <= 12)
4206                            <1>       ;;loop loc_get_fchar
4207 0000B0AB EBD0              <1>       jmp    short loc_get_fchar
4208                            <1>
4209                            <1> stop_convert_file:
4210                            <1>       ; 06/03/2016
4211 0000B0AD 30ED              <1>       xor    ch, ch
4212                            <1>       ; ECX < 256 ; 'find_first_file' -> xor cl, cl
4213                            <1> stop_convert_file_x:
4214 0000B0AF 5F                <1>       pop    edi
4215 0000B0B0 5E                <1>       pop    esi
4216 0000B0B1 C3                <1>       retn
4217                            <1>
4218                            <1> save_longname_sub_component:
4219                            <1>       ; 13/02/2016
4220                            <1>       ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
4221                            <1>       ; 28/02/2010
4222                            <1>       ; 17/10/2009
4223                            <1>       ; INPUT ->
4224                            <1>       ;      EDI = Directory Entry
4225                            <1>       ;      // This procedure is called
4226                            <1>       ;      // from 'find_directory_entry' procedure.
4227                            <1>       ;      // If the last entry returns with
4228                            <1>       ;      // a non-zero LongnameFound value and
4229                            <1>       ;      // if LFN_CheckSum value is equal to
4230                            <1>       ;      // the next shortname checksum,
4231                            <1>       ;      // long name is valid.
4232                            <1>       ;      // If a longname is longer than 65 bytes,
4233                            <1>       ;      // it is invalid for trdos. (>45h)
4234                            <1>
4235 0000B0B2 57                <1>       push   edi
4236 0000B0B3 56                <1>       push   esi
4237                            <1>       ;push  ebx
4238                            <1>       ;push  ecx
4239                            <1>       ;push  edx
4240 0000B0B4 50                <1>       push   eax
4241                            <1>
4242 0000B0B5 29C9              <1>       sub    ecx, ecx
4243                            <1>       ;sub   eax, eax
4244 0000B0B7 B11A              <1>       mov    cl, 26
4245                            <1>
4246 0000B0B9 0FB607            <1>       movzx  eax, byte [edi] ; LDIR_Order
4247 0000B0BC 3C41              <1>       cmp    al, 41h  ; 40h (last long entry sign) + 1
4248 0000B0BE 722B              <1>       jb     short pass_pslnsc_last_long_entry
4249                            <1>
4250 0000B0C0 88C4              <1>       mov    ah, al
4251 0000B0C2 80EC40            <1>       sub    ah, 40h
4252 0000B0C5 8825[6A890100]    <1>       mov    [LFN_EntryLength], ah
4253                            <1>
4254 0000B0CB 3C45              <1>       cmp    al, 45h  ; 40h (last long entry sign) + 5
4255                            <1>              ; Max 130 byte length is usable in TRDOS
4256                            <1>              ; 26*5 = 130
4257 0000B0CD 7753              <1>       ja     short loc_pslnsc_retn
4258                            <1>
4259 0000B0CF 2407              <1>       and    al, 07h ; 0Fh
4260 0000B0D1 A2[69890100]      <1>       mov    [LongNameFound], al
4261                            <1>
4262 0000B0D6 FEC8              <1>       dec    al
4263                            <1>       ;mov   cl, 26
4264 0000B0D8 F6E1              <1>       mul    cl
4265                            <1>
4266 0000B0DA 89C6              <1>       mov    esi, eax
4267 0000B0DC 01CE              <1>       add    esi, ecx
4268                            <1>              ; to make is an ASCIIZ string
4269                            <1>              ; with ax+26 bytes length
4270 0000B0DE 81C6[6C890100]    <1>       add    esi, LongFileName
4271 0000B0E4 66C7060000        <1>       mov    word [esi], 0
4272 0000B0E9 EB16              <1>       jmp    short loc_pslsc_move_ldir_name2
4273                            <1>
4274                            <1> pass_pslnsc_last_long_entry:
4275 0000B0EB 3C04              <1>       cmp    al, 04h
4276 0000B0ED 7733              <1>       ja     short loc_pslnsc_retn
4277 0000B0EF FE0D[69890100]    <1>       dec    byte [LongNameFound]
4278 0000B0F5 3A05[69890100]    <1>       cmp    al, [LongNameFound]
4279 0000B0FB 7525              <1>       jne    short loc_pslnsc_retn
4280                            <1>
4281                            <1> loc_pslsc_move_ldir_name1:
4282 0000B0FD FEC8              <1>       dec    al
4283                            <1>       ;mov   cl, 26
4284 0000B0FF F6E1              <1>       mul    cl
4285                            <1>
4286                            <1> loc_pslsc_move_ldir_name2:
4287 0000B101 8A4F0D            <1>       mov    cl, [edi+0Dh] ; long name checksum
4288 0000B104 880D[6B890100]    <1>       mov    [LFN_CheckSum], cl
```

```
4289 0000B10A 89FE                <1>       mov    esi, edi ; LDIR_Order
4290 0000B10C BF[6C890100]        <1>       mov    edi, LongFileName
4291 0000B111 01C7                <1>       add    edi, eax
4292 0000B113 46                  <1>       inc    esi
4293 0000B114 B105                <1>       mov    cl, 5 ; chars 1 to 5
4294 0000B116 F366A5              <1>       rep    movsw
4295 0000B119 83C603              <1>       add    esi, 3
4296 0000B11C A5                  <1>       movsd  ; char 6 & 7
4297 0000B11D A5                  <1>       movsd  ; char 8 & 9
4298 0000B11E A5                  <1>       movsd  ; char 10 & 11
4299 0000B11F 46                  <1>       inc    esi
4300 0000B120 46                  <1>       inc    esi
4301 0000B121 A5                  <1>       movsd   ; char 12 & 13
4302                              <1>
4303                              <1> loc_pslnsc_retn:
4304 0000B122 58                  <1>       pop    eax
4305                              <1>       ;pop   edx
4306                              <1>       ;pop   ecx
4307                              <1>       ;pop   ebx
4308 0000B123 5E                  <1>       pop    esi
4309 0000B124 5F                  <1>       pop    edi
4310                              <1>
4311 0000B125 C3                  <1>       retn
4312                              <1>
4313                              <1> parse_path_name:
4314                              <1>       ; 10/02/2016
4315                              <1>       ; 08/02/2016 (TRDOS 386 = TRDOS v2.0)
4316                              <1>       ; 10/009/2011 ('proc_parse_pathname')
4317                              <1>       ; 27/11/2009
4318                              <1>       ; 05/12/2004
4319                              <1>       ;
4320                              <1>       ; INPUT ->
4321                              <1>       ;      ESI = Beginning of ASCIIZ pathname string
4322                              <1>       ;       EDI = Destination Address
4323                              <1>       ;            (which is TR-DOS FindFile data buffer)
4324                              <1>       ; OUTPUT ->
4325                              <1>       ;      CF = 1 -> Error
4326                              <1>       ;           EAX = Error Code (AL)
4327                              <1>       ;
4328                              <1>       ; (Modified registers: eax, ecx, esi, edi)
4329                              <1>
4330                              <1>       ; Clear the pathname bytes in TR-DOS Findfile data buffer
4331 0000B126 57                  <1>       push   edi
4332 0000B127 B914000000          <1>       mov    ecx, 20  ; 80 bytes
4333 0000B12C 31C0                <1>       xor    eax, eax
4334 0000B12E F3AB                <1>       rep    stosd
4335 0000B130 5F                  <1>       pop    edi
4336                              <1>
4337 0000B131 668B06              <1>       mov    ax, [esi]
4338 0000B134 80FC3A              <1>       cmp    ah, ':'
4339 0000B137 741C                <1>       je     short loc_ppn_change_drive
4340 0000B139 A0[6E810100]        <1>       mov    al, [Current_Drv]
4341 0000B13E EB33                <1>       jmp    short pass_ppn_change_drive
4342                              <1>
4343                              <1> pass_ppn_cdir:
4344 0000B140 8B35[8E8A0100]      <1>       mov    esi, [First_Path_Pos]
4345 0000B146 AC                  <1>       lodsb
4346                              <1> loc_ppn_get_filename:
4347 0000B147 83C741              <1>       add    edi, 65 ; FindFile_Name location
4348                              <1>       ; TRDOS Filename length must not be more than 12 bytes
4349                              <1>       ;mov   ecx, 12
4350 0000B14A B10C                <1>       mov    cl, 12
4351                              <1> loc_ppn_get_fnchar_next:
4352 0000B14C AA                  <1>       stosb
4353 0000B14D AC                  <1>       lodsb
4354 0000B14E 3C21                <1>       cmp    al, 21h
4355 0000B150 7274                <1>       jb     short loc_ppn_clc_return
4356 0000B152 E2F8                <1>         loop    loc_ppn_get_fnchar_next
4357                              <1> loc_ppn_return:
4358 0000B154 C3                  <1>       retn
4359                              <1>
4360                              <1> loc_ppn_change_drive:
4361 0000B155 24DF                <1>       and    al, 0DFh
4362 0000B157 2C41                <1>       sub    al, 'A'; A:
4363 0000B159 726F                <1>       jc     short loc_ppn_invalid_drive
4364 0000B15B 3805[54390100]      <1>       cmp    [Last_DOS_DiskNo], al
4365 0000B161 7267                <1>       jb     short loc_ppn_invalid_drive
4366                              <1>
4367 0000B163 46                  <1>       inc    esi
4368 0000B164 46                  <1>       inc    esi
4369 0000B165 8A26                <1>       mov    ah, [esi]
4370 0000B167 80FC21              <1>       cmp    ah, 21h
4371 0000B16A 7307                <1>       jnb    short pass_ppn_change_drive
4372                              <1>
4373                              <1> loc_ppn_cmd_failed:
4374                              <1>       ; File or directory name is not existing
4375 0000B16C 8807                <1>       mov    [edi], al ; Drv
4376 0000B16E 66B80100            <1>       mov    ax, 1 ; eax = 1
4377                              <1>       ; TR-DOS Error Code 01h = Bad Command Argument
4378                              <1>       ; MS-DOS Error Code 01h : Invalid Function Number
4379                              <1>       ;stc
4380                              <1>       ; (MainProg ErrMsg: "Bad command or file name!")
4381 0000B172 C3                  <1>       retn
4382                              <1>
4383                              <1> pass_ppn_change_drive:
4384 0000B173 8935[8E8A0100]      <1>       mov    [First_Path_Pos], esi
4385 0000B179 C705[928A0100]0000- <1>       mov    dword [Last_Slash_Pos], 0
4385 0000B181 0000                <1>
4386 0000B183 AA                  <1>       stosb
4387 0000B184 8A06                <1>       mov    al, [esi]
4388                              <1> loc_scan_ppn_dslash:
4389 0000B186 3C2F                <1>       cmp    al, '/'
4390 0000B188 7506                <1>       jne    short loc_scan_next_slash_pos
4391 0000B18A 8935[928A0100]      <1>       mov    [Last_Slash_Pos], esi
4392                              <1> loc_scan_next_slash_pos:
```

```
4393 0000B190 46                  <1>       inc    esi
4394 0000B191 8A06                <1>       mov    al, [esi]
4395 0000B193 3C20                <1>       cmp    al, 20h
4396 0000B195 77EF                <1>       ja     short loc_scan_ppn_dslash
4397 0000B197 833D[928A0100]00    <1>       cmp    dword [Last_Slash_Pos], 0
4398 0000B19E 76A0                <1>       jna    short pass_ppn_cdir
4399                              <1>
4400 0000B1A0 8B0D[928A0100]      <1>       mov    ecx, [Last_Slash_Pos]
4401 0000B1A6 8B35[8E8A0100]      <1>       mov    esi, [First_Path_Pos]
4402 0000B1AC 29F1                <1>       sub    ecx, esi
4403 0000B1AE 41                  <1>       inc    ecx
4404                              <1>       ;cmp   ecx, 64
4405 0000B1AF 80F940              <1>       cmp    cl, 64
4406 0000B1B2 7715                <1>       ja     short loc_ppn_invalid_drive_stc
4407                              <1>
4408 0000B1B4 89F8                <1>       mov    eax, edi ; Dest Dir String Location (65 byte)
4409 0000B1B6 F3A4                <1>       rep    movsb
4410                              <1>       ;mov   [edi], cl ; 0, End of Dir String
4411 0000B1B8 8B35[928A0100]      <1>       mov    esi, [Last_Slash_Pos]
4412 0000B1BE 46                  <1>       inc    esi
4413 0000B1BF 89C7                <1>       mov    edi, eax
4414 0000B1C1 AC                  <1>       lodsb
4415 0000B1C2 3C21                <1>       cmp    al, 21h
4416 0000B1C4 7381                <1>       jnb    short loc_ppn_get_filename
4417                              <1> loc_ppn_clc_return:
4418                              <1>       ;clc
4419 0000B1C6 31C0                <1>       xor    eax, eax
4420 0000B1C8 C3                  <1>       retn
4421                              <1>
4422                              <1> loc_ppn_invalid_drive_stc:
4423 0000B1C9 F5                  <1>       cmc    ; stc
4424                              <1> loc_ppn_invalid_drive:
4425                              <1>       ; cf = 1
4426                              <1>       ; The Drive Letter/Char < "A" or > "Z"
4427 0000B1CA 66B80F00            <1>       mov    ax, 0Fh
4428                              <1>       ; MS-DOS Error Code 0Fh = Disk Drive Invalid
4429                              <1>       ; (MainProg ErrMsg: "Drive not ready or read error!")
4430 0000B1CE C3                  <1>       retn
4431                              <1>
4432                              <1> find_longname:
4433                              <1>       ; 13/02/2016 (TRDOS 386 = TRDOS v2.0)
4434                              <1>       ; 24/01/2010 (DIR.ASM, 'proc_find_longname')
4435                              <1>       ; 17/10/2009
4436                              <1>
4437                              <1>       ; INPUT ->
4438                              <1>       ;      ESI = DOS short file name address
4439                              <1>       ;      for example: "filename.ext"
4440                              <1>       ;
4441                              <1>       ; OUTPUT ->
4442                              <1>       ;      ESI = ASCIIZ longname address (cf = 0)
4443                              <1>       ;      cf = 1 -> error number returns in EAX (AL)
4444                              <1>       ;      AL = 0 & CF=1 -> longname not found
4445                              <1>       ;           the file/directory has no longname
4446                              <1>       ;      cf = 0 -> AL = FAT Type
4447                              <1>
4448                              <1>       ; 17/10/2009
4449                              <1>       ; ASCIIZ string will be returned
4450                              <1>       ; as LongFileName
4451                              <1>       ; clearing/reset is not needed
4452                              <1>       ;mov   ecx, 33
4453                              <1>       ;mov   edi, LongFileName
4454                              <1>       ;sub   ax, ax ; 0
4455                              <1>       ;rep   stosw
4456                              <1>
4457                              <1>       ;mov   byte [LongNameFound], 0
4458                              <1>
4459                              <1>       ; ESI = ASCIIZ file/directory name address
4460                              <1>       ;   AL = Attributes AND mask
4461                              <1>       ;      (Result of AND must be equal to AL)
4462                              <1>       ;   AH = Negative attributes mask
4463                              <1>       ;      (Result of AND must be ZERO)
4464 0000B1CF 66B80008            <1>       mov    ax, 0800h
4465                              <1>             ; it must not be volume name or longname
4466 0000B1D3 E87DDDFFFF          <1>       call   find_first_file
4467 0000B1D8 7216                <1>       jc     short loc_fln_retn
4468                              <1>
4469                              <1> loc_fln_check_FAT_Type:
4470 0000B1DA 803D[6D810100]01    <1>       cmp    byte [Current_FATType], 1
4471 0000B1E1 7306                <1>       jnb    short loc_fln_check_longname_yes_sign
4472                              <1>
4473 0000B1E3 E839000000          <1>       call   get_fs_longname
4474 0000B1E8 C3                  <1>       retn
4475                              <1>
4476                              <1> loc_fln_check_longname_yes_sign:
4477 0000B1E9 08FF                <1>       or     bh, bh
4478 0000B1EB 7504                <1>       jnz    short loc_fln_check_longnamefound_number
4479                              <1> loc_fln_longname_not_found_retn:
4480 0000B1ED 31C0                <1>       xor    eax, eax
4481                              <1>       ; cf = 1 & al = 0 -> longname not found
4482 0000B1EF F9                  <1>       stc
4483                              <1> loc_fln_retn:
4484 0000B1F0 C3                  <1>       retn
4485                              <1>
4486                              <1> loc_fln_check_longnamefound_number:
4487                              <1>       ; 'LongNameFound' is set by
4488                              <1>         ; by 'save_longname_sub_component'
4489                              <1>       ; which is called from
4490                              <1>       ; 'find_directory_entry'
4491                              <1>       ; which is called from
4492                              <1>       ; 'find_first_file'
4493                              <1>       ; It must 1 if the longname is valid
4494 0000B1F1 803D[69890100]01    <1>       cmp    byte [LongNameFound], 1
4495 0000B1F8 75F3                <1>       jne    short loc_fln_longname_not_found_retn
4496                              <1>
4497                              <1> loc_fln_calculate_checksum:
```

```
4498 0000B1FA E813000000          <1>       call   calculate_checksum
4499                              <1>       ; AL = shortname checksum
4500                              <1>
4501                              <1> loc_fln_longname_validation:
4502                              <1>       ; 'LFN_CheckSum' has been set already
4503                              <1>       ; by 'save_longname_sub_component'
4504                              <1>       ; which is called from
4505                              <1>       ; 'find_directory_entry'
4506                              <1>       ; which is called from
4507                              <1>       ; 'find_first_file'
4508 0000B1FF 3805[6B890100]      <1>       cmp    [LFN_CheckSum], al
4509 0000B205 75E6                <1>       jne    short loc_fln_longname_not_found_retn
4510                              <1>
4511 0000B207 BE[6C890100]        <1>       mov    esi, LongFileName
4512 0000B20C A0[6D810100]        <1>       mov    al, [Current_FATType]
4513 0000B211 C3                  <1>       retn
4514                              <1>
4515                              <1> calculate_checksum:
4516                              <1>       ; 13/02/2016 (TRDOS 386 = TRDOS v2.0)
4517                              <1>       ; 17/10/2009 (DIR.ASM, 'proc_calculate_checksum')
4518                              <1>         ;
4519                              <1>       ; INPUT ->
4520                              <1>       ;      ESI = 11 byte DOS File Name location
4521                              <1>       ;      (in DOS Directory Entry Format)
4522                              <1>       ; OUTPUT ->
4523                              <1>       ;       AL = 8 bit checksum (CRC) value
4524                              <1>       ;
4525                              <1>       ; (Modified registers: EAX, ECX, ESI)
4526                              <1>
4527                              <1>       ; Erdogan Tan [ 17-10-2009 ]
4528                              <1>       ;  'ror al, 1' instruction
4529                              <1>
4530                              <1>       ; Erdogan Tan [ 20-06-2004 ]
4531                              <1>       ; This 8086 assembly code is an original code
4532                              <1>       ; which is adapted from C code in
4533                              <1>       ; Microsoft FAT32 File System Specification
4534                              <1>       ; Version 1.03, December 6, 2000
4535                              <1>       ; Page 28
4536                              <1>
4537 0000B212 30C0                <1>       xor    al, al
4538 0000B214 B90B000000          <1>       mov    ecx, 11
4539                              <1> loc_next_sum:
4540                              <1>       ;xor   ah, ah
4541                              <1>       ;test  al, 1
4542                              <1>       ;jz    short pass_ah_80h
4543                              <1>       ;mov   ah, 80h
4544                              <1> ;pass_ah_80h:
4545                              <1>       ;shr   al, 1
4546 0000B219 D0C8                <1>       ror    al, 1 ; 17/10/2009
4547 0000B21B 0206                <1>       add    al, [esi]
4548 0000B21D 46                  <1>       inc    esi
4549                              <1>       ;add   al, ah
4550 0000B21E E2F9                <1>       loop   loc_next_sum
4551 0000B220 C3                  <1>       retn
4552                              <1>
4553                              <1> get_fs_longname:
4554                              <1>       ; temporary (13/02/2016)
4555 0000B221 31C0                <1>       xor eax, eax
4556 0000B223 F9                  <1>       stc
4557 0000B224 C3                  <1>       retn
4558                              <1>
4559                              <1> make_sub_directory:
4560                              <1>       ; 16/10/2016
4561                              <1>       ; 02/03/2016, 03/03/2016
4562                              <1>       ; 26/02/2016, 27/02/2016
4563                              <1>       ; 21/02/2016 (TRDOS 386 = TRDOS v2.0)
4564                              <1>       ; 01/08/2011 (DIR.ASM, 'proc_make_directory')
4565                              <1>       ; 10/07/2010
4566                              <1>       ; INPUT ->
4567                              <1>       ;      ESI = ASCIIZ Directory Name
4568                              <1>       ;      CL = Directory Attributes
4569                              <1>       ; OUTPUT ->
4570                              <1>       ;      EAX = New sub dir's first cluster
4571                              <1>       ;      ESI = Logical Dos Drv Descr. Table Addr.
4572                              <1>       ;      CF = 1 -> error code in AL (EAX)
4573                              <1>
4574                              <1>       ;test  cl, 10h  ; directory
4575                              <1>       ;jz    short loc_make_directory_access_denied
4576                              <1>       ;test  cl, 08h ; volume name
4577                              <1>       ;jnz   short loc_make_directory_access_denied
4578                              <1>
4579 0000B225 80E107              <1>       and    cl, 07h
4580 0000B228 880D[E88A0100]      <1>       mov    byte [mkdir_attrib], cl
4581                              <1>
4582 0000B22E 56                  <1>       push   esi
4583 0000B22F 31DB                <1>       xor    ebx, ebx
4584 0000B231 8A3D[6E810100]      <1>       mov    bh, [Current_Drv]
4585 0000B237 BE00010900          <1>       mov    esi, Logical_DOSDisks
4586 0000B23C 01DE                <1>       add    esi, ebx
4587 0000B23E 5B                  <1>       pop    ebx
4588                              <1>
4589                              <1>       ; 10/07/2010 -> 1st writable disk check for trdos
4590                              <1>       ; LD_DiskType = 0 for write protection (read only)
4591 0000B23F 807E0101            <1>       cmp    byte [esi+LD_DiskType], 1 ; 0 = Invalid
4592 0000B243 730B                <1>       jnb    short loc_mkdir_check_file_sytem
4593                              <1>       ; 16/10/2016 (13h -> 30)
4594 0000B245 B81E000000          <1>       mov    eax, 30 ; 'Disk write-protected' error
4595 0000B24A BA00000000          <1>       mov    edx, 0
4596                              <1>       ; err retn: EDX = 0, EBX = Dir name offset
4597                              <1>       ;ESI = Logical DOS drive description table address
4598 0000B24F C3                  <1>       retn
4599                              <1>
4600                              <1> ;loc_make_directory_access_denied:
4601                              <1>       ;mov   ax, 05h ; access denied (invalid attributes input)
4602                              <1>       ;stc
```

```
4603                                <1>       ;retn
4604                                <1>
4605                                <1> loc_mkdir_check_file_sytem:
4606 0000B250 807E0301             <1>       cmp    byte [esi+LD_FATType], 1
4607 0000B254 730B                 <1>       jnb    short loc_mkdir_check_free_sectors
4608                                <1>
4609                                <1> loc_make_fs_directory:
4610 0000B256 A1[68810100]         <1>       mov    eax, [Current_Dir_FCluster]
4611                                <1>       ; EAX = Parent directory DDT Address
4612                                <1>       ; ESI = Logical DOS Drive DT Address
4613                                <1>       ; EBX = Directory name offset (as ASCIIZ name)
4614 0000B25B E8D5150000           <1>       call   make_fs_directory
4615 0000B260 C3                   <1>       retn
4616                                <1>
4617                                <1> loc_mkdir_check_free_sectors:
4618 0000B261 0FB64613             <1>       movzx  eax, byte [esi+LD_BPB+SecPerClust]
4619 0000B265 8B4E74               <1>       mov    ecx, [esi+LD_FreeSectors]
4620 0000B268 39C1                 <1>       cmp    ecx, eax
4621 0000B26A 7255                 <1>       jb     short loc_mkdir_insufficient_disk_space
4622                                <1>
4623                                <1> loc_make_fat_directory:
4624 0000B26C 891D[D88A0100]       <1>       mov    [mkdir_DirName_Offset], ebx
4625 0000B272 890D[E48A0100]       <1>       mov    [mkdir_FreeSectors], ecx
4626                                <1>
4627                                <1>       ;mov   al, [esi+LD_BPB+SecPerClust]
4628 0000B278 A2[EA8A0100]         <1>       mov    byte [mkdir_SecPerClust], al
4629                                <1>
4630                                <1> loc_mkdir_gffc_1:
4631 0000B27D E80F180000           <1>       call   get_first_free_cluster
4632 0000B282 722A                 <1>       jc     short loc_mkdir_gffc_retn
4633                                <1>
4634                                <1> ;loc_mkdir_gffc_1_cont:
4635                                <1>       ;cmp   eax, 2
4636                                <1>       ;jb    short loc_mkdir_gffc_insufficient_disk_space
4637                                <1>
4638                                <1> ;loc_mkdir_gffc_1_save_fcluster:
4639 0000B284 A3[DC8A0100]         <1>       mov    [mkdir_FFCluster], eax
4640                                <1>
4641                                <1> loc_mkdir_locate_ffe:
4642                                <1>       ; Current directory fcluster <> Directory buffer cluster
4643                                <1>       ; Current directory will be reloaded by
4644                                <1>       ; 'locate_current_dir_file' procedure
4645                                <1>       ;
4646                                <1>       ; ESI = Logical DOS Drive Description Table Address
4647                                <1>       ;push  esi ; 27/02/2016
4648 0000B289 31C0                 <1>       xor    eax, eax
4649 0000B28B 89C1                 <1>         mov  ecx, eax
4650 0000B28D 6649                 <1>       dec    cx ; FFFFh
4651                                <1>       ; CX = FFFFh -> find first deleted or free entry
4652                                <1>       ; ESI would be ASCIIZ filename address if the call
4653                                <1>       ; would not be for first free or deleted dir entry
4654 0000B28F E8CFFAFFFF           <1>       call   locate_current_dir_file
4655 0000B294 734C                 <1>       jnc    short loc_mkdir_set_ff_dir_entry_1
4656                                <1>       ;pop   esi
4657                                <1>       ; ESI = Logical DOS Drive Description Table Address
4658 0000B296 83F802               <1>       cmp    eax, 2  ; cmp al, 2 ; File/Dir not found !
4659 0000B299 752B                 <1>       jne    short loc_mkdir_stc_return
4660                                <1>
4661                                <1> loc_mkdir_add_new_cluster:
4662 0000B29B 3805[6D810100]       <1>       cmp    byte [Current_FATType], al ; 2
4663                                <1>       ;cmp   byte ptr [esi+LD_FATType], 2
4664 0000B2A1 770C                 <1>       ja     short loc_mkdir_add_new_cluster_check_fsc
4665 0000B2A3 803D[6C810100]01     <1>       cmp    byte [Current_Dir_Level], 1
4666                                <1>       ;cmp   byte [esi+LD_CDirLevel], 1
4667 0000B2AA 7303                 <1>       jnb    short loc_mkdir_add_new_cluster_check_fsc
4668                                <1>
4669 0000B2AC B00C                 <1>       mov    al, 12 ; No more files
4670                                <1> loc_mkdir_gffc_retn:
4671 0000B2AE C3                   <1>       retn
4672                                <1>
4673                                <1> loc_mkdir_add_new_cluster_check_fsc:
4674 0000B2AF 8B0D[E48A0100]       <1>       mov    ecx, [mkdir_FreeSectors]
4675                                <1>       ;movzx eax, byte [mkdir_SecPerClust]
4676 0000B2B5 A0[EA8A0100]         <1>       mov    al, [mkdir_SecPerClust]
4677 0000B2BA 66D1E0               <1>       shl    ax, 1 ; AX = 2 * AX
4678 0000B2BD 39C1                 <1>       cmp    ecx, eax
4679 0000B2BF 7350                 <1>       jnb    short loc_mkdir_add_new_subdir_cluster
4680                                <1>
4681                                <1> loc_mkdir_insufficient_disk_space:
4682                                <1>       ;mov   edx, ecx
4683                                <1> ;loc_mkdir_gffc_insufficient_disk_space:
4684 0000B2C1 66B82700             <1>       mov    ax, 27h ; MSDOS err => insufficient disk space
4685                                <1>       ; err retn: EDX = Free sectors, EBX = Dir name offset
4686                                <1>         ; ESI -> Dos drive description table address
4687                                <1>       ;; ecx = edx
4688                                <1>       ;
4689 0000B2C5 C3                   <1>       retn
4690                                <1>
4691                                <1> loc_mkdir_stc_return:
4692 0000B2C6 F9                   <1>       stc
4693 0000B2C7 C3                   <1>       retn
4694                                <1>
4695                                <1> loc_mkdir_gffc_2:
4696 0000B2C8 E8C4170000           <1>       call   get_first_free_cluster
4697 0000B2CD 72DF                 <1>       jc     short loc_mkdir_gffc_retn
4698                                <1>
4699                                <1> ;loc_mkdir_gffc_1_cont:
4700                                <1>       ;cmp   eax, 2
4701                                <1>       ;jb    short loc_mkdir_gffc_insufficient_disk_space
4702                                <1>
4703                                <1> ;loc_mkdir_gffc_2_save_fcluster:
4704 0000B2CF A3[DC8A0100]         <1>       mov    [mkdir_FFCluster], eax
4705                                <1>
4706 0000B2D4 A1[E08A0100]         <1>       mov    eax, [mkdir_LastDirCluster]
4707                                <1>
```

```
4708 0000B2D9 E842170000          <1>       call   load_FAT_sub_directory
4709 0000B2DE 72CE                <1>       jc     short loc_mkdir_gffc_retn
4710                              <1>
4711 0000B2E0 31FF                <1>       xor    edi, edi
4712                              <1> loc_mkdir_set_ff_dir_entry_1:
4713                              <1>       ; 27/02/2016
4714 0000B2E2 56                  <1>       push   esi ; Logical DOS Drv Desc. Tbl. address
4715                              <1>       ; EDI = Directory Entry Address
4716 0000B2E3 8B35[D88A0100]      <1>       mov    esi, [mkdir_DirName_Offset]
4717 0000B2E9 A1[DC8A0100]        <1>       mov    eax, [mkdir_FFCluster]
4718                              <1>
4719 0000B2EE 66B91000            <1>       mov    cx, 10h ; CL = Directory attribute
4720                              <1>                      ; CH = 0 -> File size is 0
4721 0000B2F2 0A0D[E88A0100]      <1>       or     cl, [mkdir_attrib] ; S, H, R
4722 0000B2F8 E8B0010000          <1>       call   make_directory_entry
4723                              <1>
4724 0000B2FD 5E                  <1>       pop    esi
4725                              <1>
4726 0000B2FE C605[94880100]02    <1>       mov    byte [DirBuff_ValidData], 2
4727 0000B305 E880020000          <1>       call   save_directory_buffer
4728 0000B30A 0F83DA000000        <1>       jnc   loc_mkdir_set_ff_dir_entry_2
4729                              <1>
4730                              <1> loc_mkdir_return:
4731 0000B310 C3                  <1>       retn
4732                              <1>
4733                              <1> loc_mkdir_add_new_subdir_cluster:
4734 0000B311 8B15[99880100]      <1>       mov    edx, [DirBuff_Cluster]
4735 0000B317 8915[E08A0100]      <1>       mov    [mkdir_LastDirCluster], edx
4736                              <1>
4737 0000B31D A1[DC8A0100]        <1>       mov    eax, [mkdir_FFCluster]
4738 0000B322 E8F9160000          <1>       call   load_FAT_sub_directory
4739 0000B327 72E7                <1>       jc     short loc_mkdir_return
4740                              <1>       ; eax = 0
4741                              <1>       ; ecx =  directory buffer sector count (<= 128)
4742                              <1>
4743                              <1> pass_mkdir_add_new_subdir_cluster:
4744 0000B329 29FF                <1>       sub    edi, edi ; 0
4745                              <1>       ;mov   al, 128 ; double word
4746                              <1>       ;mul   ecx ; ecx =  directory buffer sector count
4747                              <1>       ;mov   ecx, eax
4748                              <1>       ;shl   cx, 7 ; 128 * sector count
4749 0000B32B 668B4611            <1>       mov    ax, [esi+LD_BPB+BytesPerSec] ; 512
4750 0000B32F 66C1E802            <1>       shr    ax, 2 ; 'byte count / 4' for 'stosd'
4751 0000B333 66F7E1              <1>       mul    cx ; max = 128*(512/4) -> 16384 (stosd)
4752 0000B336 6689C1              <1>       mov    cx, ax
4753 0000B339 6629C0              <1>       sub    ax, ax ; 0
4754 0000B33C F3AB                <1>       rep    stosd ; clear directory buffer
4755                              <1>
4756 0000B33E C605[94880100]02    <1>       mov    byte [DirBuff_ValidData], 2
4757 0000B345 E840020000          <1>       call   save_directory_buffer
4758 0000B34A 72C4                <1>       jc     short loc_mkdir_return
4759                              <1>
4760                              <1> loc_mkdir_save_added_cluster:
4761 0000B34C A1[E08A0100]        <1>       mov    eax, [mkdir_LastDirCluster]
4762 0000B351 8B0D[DC8A0100]      <1>       mov    ecx, [mkdir_FFCluster]
4763                              <1>       ; 01/03/2016
4764 0000B357 31D2                <1>       xor    edx, edx
4765 0000B359 8915[8A880100]      <1>       mov    [FAT_ClusterCounter], edx ; 0 ; reset
4766 0000B35F E800180000          <1>       call   update_cluster
4767 0000B364 7304                <1>       jnc    short loc_mkdir_save_fat_buffer_0
4768 0000B366 09C0                <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
4769 0000B368 7518                <1>       jnz    short loc_mkdir_save_fat_buffer_stc_retn
4770                              <1>
4771                              <1> loc_mkdir_save_fat_buffer_0:
4772 0000B36A A1[DC8A0100]        <1>       mov    eax, [mkdir_FFCluster]
4773 0000B36F A3[E08A0100]        <1>       mov    [mkdir_LastDirCluster], eax
4774                              <1>
4775 0000B374 31C9                <1>       xor    ecx, ecx
4776 0000B376 49                  <1>       dec    ecx ; FFFFFFFFh
4777                              <1>       ; ESI = Logical DOS Drive Description Table address
4778 0000B377 E8E8170000          <1>       call   update_cluster
4779 0000B37C 731A                <1>       jnc    short loc_mkdir_save_fat_buffer_1
4780 0000B37E 09C0                <1>       or     eax, eax
4781 0000B380 7416                <1>       jz     short loc_mkdir_save_fat_buffer_1
4782                              <1>
4783                              <1> loc_mkdir_save_fat_buffer_stc_retn:
4784                              <1>       ; 01/03/2016
4785 0000B382 803D[8A880100]01    <1>       cmp    byte [FAT_ClusterCounter], 1
4786 0000B389 720C                <1>       jb     short loc_mkdir_save_fat_buffer_retn
4787                              <1>
4788 0000B38B 66BB00FF            <1>       mov    bx, 0FF00h ; recalculate free space (BL = 0)
4789                              <1>                         ; (BH = FFh -> Use ESI as Drv Param. Tbl.)
4790 0000B38F 50                  <1>       push   eax
4791 0000B390 E8211B0000          <1>       call   calculate_fat_freespace
4792 0000B395 58                  <1>       pop    eax
4793 0000B396 F9                  <1>       stc
4794                              <1> loc_mkdir_save_fat_buffer_retn:
4795 0000B397 C3                  <1>       retn
4796                              <1>
4797                              <1> loc_mkdir_save_fat_buffer_1:
4798                              <1>       ; byte [FAT_BuffValidData] = 2
4799 0000B398 E8841A0000          <1>       call   save_fat_buffer
4800 0000B39D 72E3                <1>       jc     short loc_mkdir_save_fat_buffer_stc_retn
4801                              <1>
4802                              <1>       ; 01/03/2016
4803 0000B39F 803D[8A880100]01    <1>       cmp    byte [FAT_ClusterCounter], 1
4804 0000B3A6 721B                <1>       jb     short loc_mkdir_save_fat_buffer_2
4805                              <1>
4806                              <1>       ; ESI = Logical DOS Drive Description Table address
4807 0000B3A8 A1[8A880100]        <1>       mov    eax, [FAT_ClusterCounter]
4808 0000B3AD 66BB01FF            <1>       mov    bx, 0FF01h ; add free clusters
4809 0000B3B1 E8001B0000          <1>       call   calculate_fat_freespace
4810                              <1>
4811                              <1>       ;inc   eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
4812                              <1>       ;jnz   short loc_mkdir_save_fat_buffer_2
```

```
                                <1>
                                <1>       ; ecx > 0 -> Recalculation is needed
0000B3B6 09C9                   <1>       or    ecx, ecx
0000B3B8 7409                   <1>       jz    short loc_mkdir_save_fat_buffer_2
                                <1>
0000B3BA 66BB00FF               <1>       mov   bx, 0FF00h ; ; recalculate free space
0000B3BE E8F31A0000             <1>       call  calculate_fat_freespace
                                <1>
                                <1> loc_mkdir_save_fat_buffer_2:
0000B3C3 C605[EB8A0100]01       <1>       mov   byte [mkdir_add_new_cluster], 1
0000B3CA E9C4000000             <1>       jmp   loc_mkdir_upd_parent_dir_lmdt
                                <1>
                                <1> loc_mkdir_update_sub_dir_cluster:
0000B3CF A1[DC8A0100]           <1>       mov   eax, [mkdir_FFCluster]
0000B3D4 29C9                   <1>       sub   ecx, ecx ; 0
                                <1>       ; 01/03/2016
0000B3D6 890D[8A880100]         <1>       mov   [FAT_ClusterCounter], ecx ; 0 ; Reset
0000B3DC 49                     <1>       dec   ecx ; 0FFFFFFFFh
                                <1>
                                <1>       ; ESI = Logical DOS Drive Desciption Table address
0000B3DD E882170000             <1>       call  update_cluster
0000B3E2 7379                   <1>       jnc   short loc_mkdir_save_fat_buffer_3
0000B3E4 09C0                   <1>       or    eax, eax ; EAX = 0 -> cluster value is 0 or eocc
0000B3E6 7475                   <1>       jz    short loc_mkdir_save_fat_buffer_3
                                <1>       ; 01/03/2016
0000B3E8 EB98                   <1>       jmp   short loc_mkdir_save_fat_buffer_stc_retn
                                <1>
                                <1> loc_mkdir_set_ff_dir_entry_2:
                                <1>       ; ESI = Logical DOS Drive Description Table address
0000B3EA A1[DC8A0100]           <1>       mov   eax, [mkdir_FFCluster]
                                <1>       ; Load disk sectors as a directory cluster
0000B3EF E82C160000             <1>       call  load_FAT_sub_directory
0000B3F4 7266                   <1>       jc    short retn_make_fat_directory
                                <1>
                                <1>       ; eax = 0
                                <1>       ; ecx =  directory buffer sector count (<= 128)
                                <1>
0000B3F6 BF40000800             <1>       mov   edi, Directory_Buffer + 64 ; 26/02/2016
                                <1>
                                <1>       ; 02/03/2016
0000B3FB 668B4611               <1>       mov   ax, [esi+LD_BPB+BytesPerSec] ; 512
0000B3FF 66C1E802               <1>       shr   ax, 2 ; 'byte count / 4' for 'stosd'
0000B403 F7E1                   <1>       mul   ecx
0000B405 89C1                   <1>       mov   ecx, eax
0000B407 6629C0                 <1>       sub   ax, ax
0000B40A F3AB                   <1>       rep   stosd
                                <1>
                                <1>       ;;mov  al, 128 ; double word
                                <1>       ;;mul  ecx ; ecx =  directory buffer sector count
                                <1>       ;;mov  ecx, eax
                                <1>       ;shl   cx, 7 ; 128 * sector count
                                <1>       ;;sub  eax, eax
                                <1>       ;;sub  al, al ; 0
                                <1>       ;rep   stosd ; clear directory buffer
                                <1>
0000B40C BF00000800             <1>       mov   edi, Directory_Buffer ; 26/02/2016
                                <1>
0000B411 56                     <1>       push  esi
                                <1>
0000B412 BE[EC8A0100]           <1>       mov   esi, mkdir_Name
0000B417 66C7062E00             <1>       mov   word [esi], 2Eh ; db '.', '0'
                                <1>
0000B41C A1[DC8A0100]           <1>       mov   eax, [mkdir_FFCluster]
0000B421 66B91000               <1>       mov   cx, 10h ; CL = Directory attribute
                                <1>                     ; CH = 0 -> File size is 0
0000B425 E883000000             <1>       call  make_directory_entry
                                <1>
0000B42A BF20000800             <1>       mov   edi, Directory_Buffer + 32 ; 26/02/2016
                                <1>
                                <1>       ; 03/03/2016
                                <1>       ; Following modification has been done according to
                                <1>       ; 'Microsoft Extensible Firmware Initiative
                                <1>       ; FAT32 File System Specification' document,
                                <1>       ; 'FAT: General Overview of On-Disk Format—Page 25'.
                                <1>       ; "Finally, you set DIR_FstClusLO and DIR_FstClusHI
                                <1>       ; for the dotdot entry (the second entry) to the
                                <1>       ; first cluster number of the directory in which you
                                <1>       ; just created the directory (value is 0 if this directory
                                <1>       ; is the root directory even for FAT32 volumes)."
                                <1>       ; (Correctness of this modification has been verified
                                <1>       ;  by using Windows 98 'scandisk.exe'.)
                                <1>
0000B42F 29C0                   <1>       sub   eax, eax
0000B431 3805[6C810100]         <1>       cmp   byte [Current_Dir_Level], al ; 0
0000B437 7605                   <1>       jna   short loc_mkdir_set_ff_dir_entry_3
0000B439 A1[68810100]           <1>       mov   eax, [Current_Dir_FCluster] ; parent dir
                                <1> loc_mkdir_set_ff_dir_entry_3:
0000B43E 66C746012E00           <1>       mov   word [esi+1], 2Eh ; db '.', '0'
                                <1>
                                <1>       ;mov  cx, 10h
0000B444 E864000000             <1>       call  make_directory_entry
                                <1>
0000B449 5E                     <1>       pop   esi
                                <1>
0000B44A C605[94880100]02       <1>       mov   byte [DirBuff_ValidData], 2
0000B451 E834010000             <1>       call  save_directory_buffer
0000B456 0F8373FFFFFF           <1>       jnc   loc_mkdir_update_sub_dir_cluster
                                <1>
                                <1> retn_make_fat_directory:
0000B45C C3                     <1>       retn
                                <1>
                                <1> loc_mkdir_save_fat_buffer_3:
                                <1>       ; 01/03/2016
                                <1>       ; byte [FAT_BuffValidData] = 2
0000B45D E8BF190000             <1>       call  save_fat_buffer
```

```
4918 0000B462 0F821AFFFFFF        <1>       jc    loc_mkdir_save_fat_buffer_stc_retn
4919                              <1>
4920 0000B468 803D[8A880100]01    <1>       cmp   byte [FAT_ClusterCounter], 1
4921 0000B46F 721B                <1>       jb    short loc_mkdir_save_fat_buffer_4
4922                              <1>
4923                              <1>       ; ESI = Logical DOS Drive Description Table address
4924 0000B471 A1[8A880100]        <1>       mov   eax, [FAT_ClusterCounter]
4925 0000B476 66BB01FF            <1>       mov   bx, 0FF01h ; add free clusters
4926 0000B47A E8371A0000          <1>       call  calculate_fat_freespace
4927                              <1>
4928                              <1>       ;inc   eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
4929                              <1>       ;jnz  short loc_mkdir_save_fat_buffer_4
4930                              <1>
4931                              <1>       ; ecx > 0 -> Recalculation is needed
4932 0000B47F 09C9                <1>       or    ecx, ecx
4933 0000B481 7409                <1>       jz    short loc_mkdir_save_fat_buffer_4
4934                              <1>
4935 0000B483 66BB00FF            <1>       mov   bx, 0FF00h ; recalculate free space
4936 0000B487 E82A1A0000          <1>       call  calculate_fat_freespace
4937                              <1>
4938                              <1> loc_mkdir_save_fat_buffer_4:
4939 0000B48C C605[EB8A0100]00    <1>       mov   byte [mkdir_add_new_cluster], 0
4940                              <1>
4941                              <1> loc_mkdir_upd_parent_dir_lmdt:
4942 0000B493 E88D010000          <1>       call  update_parent_dir_lmdt
4943                              <1>
4944                              <1>       ; 01/03/2016
4945 0000B498 803D[EB8A0100]00    <1>       cmp   byte [mkdir_add_new_cluster], 0
4946 0000B49F 0F8723FEFFFF        <1>       ja    loc_mkdir_gffc_2
4947                              <1>
4948                              <1> loc_mkdir_retn_new_dir_cluster:
4949 0000B4A5 A1[DC8A0100]        <1>       mov   eax, [mkdir_FFCluster]
4950 0000B4AA 31D2                <1>       xor   edx, edx
4951                              <1> loc_mkdir_retn:
4952 0000B4AC C3                  <1>       retn
4953                              <1>
4954                              <1> make_directory_entry:
4955                              <1>       ; 02/03/2016
4956                              <1>       ; 21/02/2016 (TRDOS 386 = TRDOS v2.0)
4957                              <1>       ; 09/08/2010 (DIR.ASM, 'proc_make_directory_entry')
4958                              <1>       ; 17/07/2010
4959                              <1>       ; INPUT ->
4960                              <1>       ;      EDI = Directory Entry Address
4961                              <1>       ;      ESI = Dot File Name Location
4962                              <1>       ;      EAX = First Cluster
4963                              <1>       ;      File Size = 0 (Must be set later)
4964                              <1>       ;      CL = Attributes
4965                              <1>       ;      CH = 0 (File size = 0)
4966                              <1>       ;      (If CH>0, File size is in dword [EBX]) (*)
4967                              <1>       ; OUTPUT ->
4968                              <1>       ;      EDI = Directory Entry Address
4969                              <1>       ;      ESI = Dot File Name Location (Capitalized)
4970                              <1>       ;      If CH input = 0, File Size = 0
4971                              <1>       ;      Otherwise file size is as dword [EBX] (*)
4972                              <1>       ;      DX = Date, AX = Time in DOS Dir Entry format
4973                              <1>       ;      EBX = same
4974                              <1>       ;      ECX = same
4975                              <1>
4976 0000B4AD 51                  <1>       push  ecx
4977                              <1>
4978 0000B4AE 884F0B              <1>       mov   [edi+11], cl ; Attributes
4979 0000B4B1 6689471A            <1>       mov   [edi+26], ax ; FClusterLw, 26
4980 0000B4B5 C1E810              <1>       shr   eax, 16
4981 0000B4B8 66894714            <1>       mov   [edi+20], ax ; FClusterHw, 20
4982 0000B4BC 6631C0              <1>       xor   ax, ax
4983 0000B4BF 6689470C            <1>       mov   [edi+12], ax ; NTReserved, 12
4984                              <1>                          ; CrtTimeTenth, 13
4985 0000B4C3 08ED                <1>       or    ch, ch
4986 0000B4C5 7402                <1>       jz    short loc_make_direntry_set_filesize
4987                              <1>
4988 0000B4C7 8B03                <1>       mov   eax, [ebx]
4989                              <1>
4990                              <1> loc_make_direntry_set_filesize:
4991 0000B4C9 89471C              <1>       mov   [edi+28], eax ; FileSize, 28
4992                              <1>
4993 0000B4CC E88AFBFFFF          <1>       call  convert_file_name
4994                              <1>       ;EDI = Dir Entry Format File Name Location
4995                              <1>       ;ESI = Dot File Name Location (capitalized)
4996                              <1>
4997 0000B4D1 E816000000          <1>       call  convert_current_date_time
4998                              <1>       ; OUTPUT -> DX = Date in dos dir entry format
4999                              <1>         ;       AX = Time in dos dir entry format
5000 0000B4D6 6689470E            <1>       mov   [edi+14], ax ; CrtTime, 14
5001 0000B4DA 66895710            <1>       mov   [edi+16], dx ; CrtDate, 16
5002 0000B4DE 66895712            <1>       mov   [edi+18], dx ; LastAccDate, 18
5003 0000B4E2 66894716            <1>       mov   [edi+22], ax ; WrtTime, 14
5004 0000B4E6 66895718            <1>       mov   [edi+24], dx ; WrtDate, 16
5005 0000B4EA 59                  <1>       pop   ecx
5006                              <1>
5007 0000B4EB C3                  <1>       retn
5008                              <1>
5009                              <1> convert_current_date_time:
5010                              <1>       ; 21/02/2016 (TRDOS 386 = TRDOS v2.0)
5011                              <1>       ; 13/06/2010 (DIR.ASM, 'proc_convert_current_date_time')
5012                              <1>       ; converts date&time to dos dir entry format
5013                              <1>       ; INPUT -> none
5014                              <1>       ; OUTPUT -> DX = Date in dos dir entry format
5015                              <1>       ;           AX = Time in dos dir entry format
5016                              <1>
5017 0000B4EC B404                <1>       mov   ah, 04h ; Return Current Date
5018 0000B4EE E840B0FFFF          <1>       call  int1Ah
5019                              <1>
5020 0000B4F3 88E8                <1>       mov   al, ch ; <- century BCD
5021 0000B4F5 240F                <1>       and   al, 0Fh
5022 0000B4F7 88EC                <1>       mov   ah, ch
```

```
5023 0000B4F9 C0EC04              <1>       shr    ah, 4
5024 0000B4FC D50A                <1>       aad
5025 0000B4FE 88C5                <1>       mov    ch, al ; -> century
5026                              <1>
5027 0000B500 88C8                <1>       mov    al, cl ; <- year BCD
5028 0000B502 240F                <1>       and    al, 0Fh
5029 0000B504 88CC                <1>       mov    ah, cl
5030 0000B506 C0EC04              <1>       shr    ah, 4
5031 0000B509 D50A                <1>       aad
5032 0000B50B 88C1                <1>       mov    cl, al ; -> year
5033                              <1>
5034 0000B50D 88E8                <1>       mov    al, ch
5035 0000B50F B464                <1>       mov    ah, 100
5036 0000B511 F6E4                <1>       mul    ah
5037 0000B513 30ED                <1>       xor    ch, ch
5038 0000B515 6601C8              <1>       add    ax, cx
5039 0000B518 662DBC07            <1>       sub    ax, 1980 ; ms-dos epoch
5040 0000B51C 6689C1              <1>       mov    cx, ax
5041                              <1>
5042 0000B51F 88F0                <1>       mov    al, dh ; <- month in bcd
5043 0000B521 240F                <1>       and    al, 0Fh
5044 0000B523 88F4                <1>       mov    ah, dh
5045 0000B525 C0EC04              <1>       shr    ah, 4
5046 0000B528 D50A                <1>       aad
5047 0000B52A 88C6                <1>       mov    dh, al ; -> month
5048                              <1>
5049 0000B52C 88D0                <1>       mov    al, dl ; <- day BCD
5050 0000B52E 240F                <1>       and    al, 0Fh
5051 0000B530 88D4                <1>       mov    ah, dl
5052 0000B532 C0EC04              <1>       shr    ah, 4
5053 0000B535 D50A                <1>       aad
5054 0000B537 88C2                <1>       mov    dl, al ; -> day
5055                              <1>
5056 0000B539 88C8                <1>       mov    al, cl ; count of years from 1980
5057 0000B53B 66C1E004            <1>       shl    ax, 4
5058 0000B53F 08F0                <1>       or     al, dh ; month of year, 1 to 12
5059 0000B541 66C1E005            <1>       shl    ax, 5
5060 0000B545 08D0                <1>       or     al, dl ; day of year, 1 to 31
5061                              <1>
5062 0000B547 6650                <1>       push   ax ; push date
5063                              <1>
5064 0000B549 B402                <1>       mov    ah, 02h ; Return Current Time
5065 0000B54B E8E3AFFFFF          <1>       call   int1Ah
5066                              <1>
5067 0000B550 88E8                <1>       mov    al, ch ; <- hours BCD
5068 0000B552 240F                <1>       and    al, 0Fh
5069 0000B554 88EC                <1>       mov    ah, ch
5070 0000B556 C0EC04              <1>       shr    ah, 4
5071 0000B559 D50A                <1>       aad
5072 0000B55B 88C5                <1>       mov    ch, al ; -> hours
5073                              <1>
5074 0000B55D 88C8                <1>       mov    al, cl ; <- minutes BCD
5075 0000B55F 240F                <1>       and    al, 0Fh
5076 0000B561 88CC                <1>       mov    ah, cl
5077 0000B563 C0EC04              <1>       shr    ah, 4
5078 0000B566 D50A                <1>       aad
5079 0000B568 88C1                <1>       mov    cl, al ; -> minutes
5080                              <1>
5081 0000B56A 88F0                <1>       mov    al, dh ; <- seconds BCD
5082 0000B56C 240F                <1>       and    al, 0Fh
5083 0000B56E 88F4                <1>       mov    ah, dh
5084 0000B570 C0EC04              <1>       shr    ah, 4
5085 0000B573 D50A                <1>       aad
5086 0000B575 88C6                <1>       mov    dh, al ; -> seconds
5087                              <1>
5088 0000B577 88E8                <1>       mov    al, ch ; hours
5089 0000B579 66C1E006            <1>       shl    ax, 6
5090 0000B57D 08C8                <1>       or     al, cl ; minutes
5091 0000B57F 66C1E005            <1>       shl    ax, 5
5092 0000B583 D0EE                <1>       shr    dh, 1 ; 2 seconds
5093                              <1>       ; There is a bug in TRDOS v1 here !
5094                              <1>       ; it was 'or al, dl' !
5095 0000B585 08F0                <1>       or     al, dh ; seconds
5096                              <1>
5097 0000B587 665A                <1>       pop    dx ; pop date
5098                              <1>
5099 0000B589 C3                  <1>       retn
5100                              <1>
5101                              <1> save_directory_buffer:
5102                              <1>       ; 15/10/2016
5103                              <1>       ; 23/03/2016
5104                              <1>       ; 26/02/2016
5105                              <1>       ; 22/02/2016 (TRDOS 386 = TRDOS v2.0)
5106                              <1>       ; 01/08/2011
5107                              <1>       ; 14/03/2010
5108                              <1>       ; INPUT ->
5109                              <1>       ;       none
5110                              <1>       ; OUTPUT ->
5111                              <1>       ;  cf = 0 -> write OK...
5112                              <1>       ;  cf = 1 -> error code in AL (EAX)
5113                              <1>       ;  cf = 1 & AL = 0Dh => CH & CL = FS & FAT type
5114                              <1>       ;  EBX = Directory Buffer Address
5115                              <1>       ;
5116                              <1>       ;  (EAX, ECX, EDX will be modified)
5117                              <1>
5118 0000B58A BB00000800          <1>       mov    ebx, Directory_Buffer
5119 0000B58F 803D[94880100]02    <1>       cmp    byte [DirBuff_ValidData], 2
5120 0000B596 7403                <1>       je     short loc_save_dir_buffer
5121 0000B598 31C0                <1>       xor    eax, eax
5122 0000B59A C3                  <1>       retn
5123                              <1>
5124                              <1> loc_save_dir_buffer:
5125 0000B59B 56                  <1>       push   esi
5126 0000B59C 31DB                <1>       xor    ebx, ebx
5127 0000B59E 8A3D[92880100]      <1>       mov   bh, [DirBuff_DRV]
```

```
5128 0000B5A4 80EF41              <1>       sub    bh, 'A'
5129 0000B5A7 BE00010900          <1>       mov   esi, Logical_DOSDisks
5130 0000B5AC 01DE                <1>       add    esi, ebx
5131 0000B5AE 668B4E03            <1>       mov   cx, [esi+LD_FATType]
5132                              <1>       ; CH = FS Type (A1h for FS)
5133                              <1>       ; CL = FAT Type (0 for FS)
5134 0000B5B2 08C9                <1>       or     cl, cl
5135 0000B5B4 7433                <1>       jz     short loc_save_dir_buff_stc_retn
5136                              <1>
5137                              <1> loc_save_dir_buffer_check_cluster_no:
5138 0000B5B6 A1[99880100]        <1>       mov    eax, [DirBuff_Cluster]
5139 0000B5BB 28FF                <1>       sub    bh, bh ; ebx = 0
5140 0000B5BD 09C0                <1>       or     eax, eax
5141 0000B5BF 7540                <1>       jnz    short loc_save_sub_dir_buffer
5142 0000B5C1 8A25[93880100]      <1>       mov    ah, [DirBuff_FATType]
5143 0000B5C7 FEC3                <1>       inc    bl ;  bl = 1
5144 0000B5C9 38DC                <1>       cmp    ah, bl
5145 0000B5CB 721D                <1>       jb     short loc_save_dir_buff_inv_data_retn
5146 0000B5CD FEC3                <1>       inc    bl ; bl = 2
5147 0000B5CF 38E3                <1>       cmp    bl, ah
5148 0000B5D1 7217                <1>       jb     short loc_save_dir_buff_inv_data_retn
5149                              <1>
5150                              <1> loc_save_root_dir_buffer:
5151 0000B5D3 668B5E17            <1>       mov    bx, [esi+LD_BPB+RootDirEnts]
5152 0000B5D7 6683C30F            <1>       add    bx, 15
5153 0000B5DB 66C1EB04            <1>       shr    bx, 4 ; 16 dir entries per sector
5154 0000B5DF 6609DB              <1>       or     bx, bx
5155 0000B5E2 7405                <1>       jz     short loc_save_dir_buff_stc_retn
5156                              <1>       ;mov   ecx, ebx
5157 0000B5E4 8B4664              <1>       mov    eax, [esi+LD_ROOTBegin] ; 26/02/2016
5158 0000B5E7 EB23                <1>       jmp    short loc_write_directory_to_disk
5159                              <1>
5160                              <1> loc_save_dir_buff_stc_retn:
5161 0000B5E9 F9                  <1>       stc
5162                              <1> loc_save_dir_buff_inv_data_retn:
5163                              <1>       ; 15/10/2016 (0Dh -> 29)
5164 0000B5EA B01D                <1>       mov    al, 29 ; Invalid data !
5165 0000B5EC C605[94880100]00    <1>       mov    byte [DirBuff_ValidData], 0
5166 0000B5F3 EB05                <1>       jmp    short loc_save_dir_buff_retn
5167                              <1>
5168                              <1> loc_write_directory_to_disk_err:
5169                              <1>       ; 15/10/2016 (disk write error code, 1Dh -> 18)
5170 0000B5F5 B812000000          <1>       mov    eax, 18 ; Drive not ready or write error
5171                              <1>
5172                              <1> loc_save_dir_buff_retn:
5173 0000B5FA BB00000800          <1>       mov    ebx, Directory_Buffer
5174 0000B5FF 5E                  <1>       pop    esi
5175 0000B600 C3                  <1>       retn
5176                              <1>
5177                              <1> loc_save_sub_dir_buffer:
5178                              <1>       ; ebx  = 0
5179 0000B601 83E802              <1>       sub    eax, 2
5180 0000B604 8A5E13              <1>       mov    bl, [esi+LD_BPB+SecPerClust]
5181 0000B607 F7E3                <1>       mul    ebx
5182 0000B609 034668              <1>       add   eax, [esi+LD_DATABegin]
5183                              <1>       ;mov   ecx, ebx
5184                              <1>
5185                              <1> loc_write_directory_to_disk:
5186 0000B60C 89D9                <1>       mov    ecx, ebx
5187 0000B60E BB00000800          <1>       mov    ebx, Directory_Buffer
5188 0000B613 E8746F0000          <1>       call   disk_write
5189 0000B618 72DB                <1>       jc     short loc_write_directory_to_disk_err
5190                              <1>
5191                              <1> loc_save_dir_buff_validate_retn:
5192 0000B61A C605[94880100]01    <1>       mov    byte [DirBuff_ValidData], 1
5193 0000B621 31C0                <1>       xor    eax, eax
5194                              <1>       ; 26/02/2016
5195 0000B623 EBD5                <1>       jmp    short loc_save_dir_buff_retn
5196                              <1>
5197                              <1> update_parent_dir_lmdt:
5198                              <1>       ; 29/12/2017
5199                              <1>       ; 22/02/2016 (TRDOS 386 = TRDOS v2.0)
5200                              <1>       ; 01/08/2011
5201                              <1>       ; 16/10/2010
5202                              <1>       ;
5203                              <1>       ; INPUT ->
5204                              <1>       ;      none
5205                              <1>       ; OUTPUT ->
5206                              <1>       ;      (last modification date & time of the parent dir
5207                              <1>       ;      will be changed/updated)
5208                              <1>       ;
5209                              <1>       ; (EAX, EBX, ECX, EDX, EDI will be changed)
5210                              <1>
5211 0000B625 29C0                <1>       sub    eax, eax
5212 0000B627 8A25[6C810100]      <1>       mov    ah, [Current_Dir_Level]
5213 0000B62D A0[6D810100]        <1>       mov    al, [Current_FATType]
5214 0000B632 3C01                <1>       cmp    al, 1
5215 0000B634 723A                <1>       jb     short loc_UPDLMDT_proc_retn
5216                              <1>
5217                              <1> loc_update_parent_dir_lm_date_time:
5218 0000B636 08E4                <1>       or     ah, ah
5219 0000B638 7436                <1>       jz     short loc_UPDLMDT_proc_retn
5220                              <1>
5221 0000B63A 56                  <1>       push   esi ; *
5222 0000B63B 8825[0C8B0100]      <1>       mov    [UPDLMDT_CDirLevel], ah
5223 0000B641 8B15[68810100]      <1>       mov    edx, [Current_Dir_FCluster]
5224 0000B647 8915[0D8B0100]      <1>       mov    [UPDLMDT_CDirFCluster], edx
5225                              <1>
5226 0000B64D FECC                <1>       dec    ah
5227 0000B64F B90C000000          <1>       mov    ecx, 12
5228 0000B654 BE[CB880100]        <1>       mov   esi, PATH_Array
5229                              <1>
5230 0000B659 8825[6C810100]      <1>       mov    [Current_Dir_Level], ah
5231 0000B65F 08E4                <1>       or     ah, ah
5232 0000B661 750E                <1>       jnz    short loc_update_parent_dir_lmdt_load_sub_dir_1
```

```
5233 0000B663 803D[6D810100]02    <1>       cmp    byte [Current_FATType], 2
5234 0000B66A 770B                <1>       ja     short loc_update_parent_dir_lmdt_load_sub_dir_2
5235 0000B66C 28C0                <1>       sub    al, al ; eax = 0
5236 0000B66E EB0A                <1>       jmp    short loc_update_parent_dir_lmdt_load_sub_dir_3
5237                              <1>
5238                              <1> loc_UPDLMDT_proc_retn:
5239 0000B670 C3                  <1>       retn
5240                              <1>
5241                              <1> loc_update_parent_dir_lmdt_load_sub_dir_1:
5242 0000B671 B010                <1>       mov    al, 16
5243 0000B673 F6E4                <1>       mul    ah
5244 0000B675 01C6                <1>       add    esi, eax
5245                              <1>
5246                              <1> loc_update_parent_dir_lmdt_load_sub_dir_2:
5247 0000B677 8B460C              <1>       mov    eax, [esi+12] ; Parent Dir First Cluster
5248                              <1>
5249                              <1> loc_update_parent_dir_lmdt_load_sub_dir_3:
5250 0000B67A A3[68810100]        <1>       mov    [Current_Dir_FCluster], eax
5251                              <1>
5252 0000B67F 83C610              <1>       add    esi, 16
5253 0000B682 66BF[F289]          <1>       mov    di, Dir_File_Name
5254 0000B686 F3A4                <1>       rep    movsb
5255                              <1>
5256 0000B688 BE00010900          <1>       mov    esi, Logical_DOSDisks
5257 0000B68D 29DB                <1>       sub    ebx, ebx
5258 0000B68F 8A3D[6E810100]      <1>       mov    bh, [Current_Drv]
5259 0000B695 01DE                <1>       add    esi, ebx
5260 0000B697 E88EF7FFFF          <1>       call   reload_current_directory
5261 0000B69C 7230                <1>       jc     short loc_update_parent_dir_lmdt_restore_cdirlevel
5262                              <1>
5263                              <1> loc_update_parent_dir_lmdt_locate_dir:
5264 0000B69E BE[F2890100]        <1>       mov    esi, Dir_File_Name
5265 0000B6A3 6631C9              <1>       xor    cx, cx
5266 0000B6A6 66B81008            <1>       mov    ax, 0810h ; Only directories
5267 0000B6AA E8B4F6FFFF          <1>         call    locate_current_dir_file
5268                              <1>       ; EDI = DirBuff Directory Entry Address
5269 0000B6AF 721D                <1>       jc     short loc_update_parent_dir_lmdt_restore_cdirlevel
5270                              <1>
5271 0000B6B1 E836FEFFFF          <1>       call   convert_current_date_time
5272 0000B6B6 66895712            <1>       mov    [edi+18], dx ; Last Access Date
5273 0000B6BA 66895718            <1>       mov    [edi+24], dx ; Last Write Date
5274 0000B6BE 66894716            <1>       mov    [edi+22], ax ; Last Write Time
5275                              <1>
5276 0000B6C2 C605[94880100]02    <1>       mov    byte [DirBuff_ValidData], 2
5277 0000B6C9 E8BCFEFFFF          <1>       call   save_directory_buffer
5278                              <1>       ; 29/12/2017
5279                              <1>       ;jc    short loc_update_parent_dir_lmdt_restore_cdirlevel
5280                              <1>       ;xor   al, al
5281                              <1> loc_update_parent_dir_lmdt_restore_cdirlevel:
5282                              <1>       ;current directory level restoration
5283 0000B6CE 8A25[0C8B0100]      <1>       mov    ah, [UPDLMDT_CDirLevel]
5284 0000B6D4 8825[6C810100]      <1>       mov    [Current_Dir_Level], ah
5285 0000B6DA 8B15[0D8B0100]      <1>       mov    edx, [UPDLMDT_CDirFCluster]
5286 0000B6E0 8915[68810100]      <1>       mov    [Current_Dir_FCluster], edx
5287                              <1>
5288 0000B6E6 5E                  <1>       pop    esi ; *
5289 0000B6E7 C3                  <1>       retn
5290                              <1>
5291                              <1> delete_longname:
5292                              <1>       ; 27/02/2016 (TRDOS 386 = TRDOS v2.0)
5293                              <1>       ; 01/08/2011 (DIR.ASM, 'proc_delete_longname')
5294                              <1>       ; 14/03/2010
5295                              <1>       ; INPUT ->
5296                              <1>       ;      EAX = Directory Entry (Index) Number (< 65536)
5297                              <1>       ; OUTPUT ->
5298                              <1>       ;      cf = 0 -> OK  (EAX = 0)
5299                              <1>       ;      cf = 1 -> error code in EAX (AL)
5300                              <1>       ;
5301                              <1>       ; (Modified registers: EAX, EDX, ECX, EBX, EDI)
5302                              <1>
5303 0000B6E8 66A3[3C8B0100]      <1>       mov    [DLN_EntryNumber], ax
5304 0000B6EE C605[3E8B0100]40    <1>       mov    byte [DLN_40h], 40h
5305                              <1>
5306 0000B6F5 E858000000          <1>       call   locate_current_dir_entry
5307 0000B6FA 7308                <1>       jnc    short loc_dln_check_attributes
5308 0000B6FC C3                  <1>       retn
5309                              <1>
5310                              <1> loc_dln_longname_not_found:
5311 0000B6FD B802000000          <1>       mov    eax, 2
5312 0000B702 F9                  <1>       stc
5313 0000B703 C3                  <1>       retn
5314                              <1>
5315                              <1> loc_dln_check_attributes:
5316 0000B704 B00F                <1>       mov    al, 0Fh  ; long name
5317 0000B706 8A670B              <1>       mov    ah, [edi+0Bh] ; dir entry attributes
5318 0000B709 38C4                <1>       cmp    ah, al
5319 0000B70B 75F0                <1>       jne    short loc_dln_longname_not_found
5320 0000B70D 8A27                <1>       mov    ah, [edi]
5321 0000B70F 2A25[3E8B0100]      <1>       sub    ah, [DLN_40h]
5322 0000B715 76E6                <1>       jna    short loc_dln_longname_not_found
5323 0000B717 80FC14              <1>       cmp    ah, 14h ; 84-64=20 -> 20*13=260 bytes
5324 0000B71A 77E1                <1>       ja     short loc_dln_longname_not_found
5325                              <1>
5326 0000B71C C607E5              <1>       mov    byte [edi], 0E5h  ; deleted sign
5327 0000B71F C605[94880100]02    <1>       mov    byte [DirBuff_ValidData], 2 ; changed/write sign
5328 0000B726 C605[3E8B0100]00    <1>       mov    byte [DLN_40h], 0 ; 40h -> 0
5329                              <1>
5330                              <1> loc_dln_delete_next_ln_entry:
5331 0000B72D 80FC01              <1>       cmp    ah, 1
5332 0000B730 7616                <1>       jna    short loc_dln_longname_retn
5333                              <1> loc_dln_delete_next_ln_entry_0:
5334 0000B732 66FF05[3C8B0100]    <1>       inc    word [DLN_EntryNumber]
5335 0000B739 0FB705[3C8B0100]    <1>       movzx  eax, word [DLN_EntryNumber]
5336 0000B740 E80D000000          <1>       call   locate_current_dir_entry
5337 0000B745 73BD                <1>       jnc    short loc_dln_check_attributes
```

```
                                <1>
                                <1> loc_dln_longname_stc_retn:
0000B747 C3                     <1>     retn
                                <1>
                                <1> loc_dln_longname_retn:
                                <1>     ;cmp   byte [DirBuff_ValidData], 2
                                <1>     ;jne   short loc_dln_longname_retn_xor_eax
0000B748 E83DFEFFFF             <1>     call   save_directory_buffer
0000B74D 72F8                   <1>     jc     short loc_dln_longname_stc_retn
                                <1>
                                <1> loc_dln_longname_retn_xor_eax:
0000B74F 31C0                   <1>     xor    eax, eax
0000B751 C3                     <1>     retn
                                <1>
                                <1> locate_current_dir_entry:
                                <1>     ; 16/10/2016
                                <1>     ; 15/10/2016
                                <1>     ; 23/03/2016
                                <1>     ; 27/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>     ; 01/08/2011 (DIR.ASM, 'proc_locate_current_dir_entry')
                                <1>     ; 07/03/2010
                                <1>     ; INPUT ->
                                <1>     ;      EAX = Directory Entry (Index) Number (< 65536)
                                <1>     ; OUTPUT ->
                                <1>     ;      EDI = Directory Entry Address
                                <1>     ;      EAX = Cluster Number of Directory Buffer
                                <1>     ;      EBX = Directory Buffer Entry Offset
                                <1>     ;      ECX = DirBuff Valid Data identifier (CL)
                                <1>     ;      If CF = 0 and CL = 2 then
                                <1>     ;         directory buffer modified and
                                <1>     ;         must be written to disk.
                                <1>     ;      If CF = 0  and CL = 1 then
                                <1>     ;         dir buffer has been written to disk, already.
                                <1>     ;      CF = 1 -> Error code in EAX (AL)
                                <1>     ;
                                <1>     ; (Modified registers: EAX, EDX, ECX, EBX, EDI)
                                <1>
                                <1> loc_locate_current_dir_entry:
0000B752 56                     <1>     push   esi
0000B753 89C1                   <1>     mov    ecx, eax
0000B755 BA20000000             <1>     mov    edx, 32
0000B75A F7E2                   <1>     mul    edx
0000B75C A3[488B0100]           <1>     mov    [LCDE_ByteOffset], eax
0000B761 31DB                   <1>     xor    ebx, ebx
0000B763 8A3D[6E810100]         <1>     mov    bh, [Current_Drv]
0000B769 A0[92880100]           <1>     mov    al, [DirBuff_DRV]
0000B76E 2C41                   <1>     sub    al, 'A'
0000B770 BE00010900             <1>     mov    esi, Logical_DOSDisks
0000B775 01DE                   <1>     add    esi, ebx
0000B777 38C7                   <1>     cmp    bh, al
0000B779 0F8592000000           <1>     jne    loc_lcde_reload_current_directory
                                <1> loc_lcde_cdl_check:
0000B77F 803D[6C810100]00       <1>     cmp    byte [Current_Dir_Level], 0
0000B786 772A                   <1>     ja     short loc_lcde_calc_dirbuff_cluster_offset
                                <1>     ; 27/02/2016
                                <1>     ; TRDOS v1 has bug here for FAT32 fs !
                                <1>     ; (Root Directory Entries for FAT32 = 0)
0000B788 807E0303               <1>     cmp    byte [esi+LD_FATType], 3  ; FAT32
0000B78C 7324                   <1>     jnb    short loc_lcde_calc_dirbuff_cluster_offset
                                <1>
                                <1> loc_lcde_cdl_check_FAT12_16:
0000B78E 668B4617               <1>     mov    ax, [esi+LD_BPB+RootDirEnts]
0000B792 6648                   <1>     dec    ax
                                <1>     ;xor   dx, dx
0000B794 6639C8                 <1>     cmp    ax, cx ; cx = Directory Entry (Index) Number
0000B797 720E                   <1>     jb     short loc_lcde_stc_12h_retn
0000B799 66890D[408B0100]       <1>     mov    [LCDE_EntryIndex], cx
0000B7A0 31C0                   <1>     xor    eax, eax
0000B7A2 E993000000             <1>     jmp    loc_lcde_check_dir_buffer_cluster
                                <1>
                                <1> loc_lcde_stc_12h_retn:
0000B7A7 5E                     <1>     pop    esi
0000B7A8 89CB                   <1>     mov    ebx, ecx
0000B7AA 89D1                   <1>     mov    ecx, edx
                                <1>     ; 16/10/2016 (12h -> 12)
0000B7AC B80C000000             <1>     mov    eax, 12 ; No more files
0000B7B1 C3                     <1>     retn
                                <1>
                                <1> loc_lcde_calc_dirbuff_cluster_offset:
0000B7B2 8A5E13                 <1>     mov    bl, [esi+LD_BPB+SecPerClust]
0000B7B5 30FF                   <1>     xor    bh, bh
0000B7B7 668B4611               <1>     mov    ax, [esi+LD_BPB+BytesPerSec]
0000B7BB 66F7E3                 <1>     mul    bx
0000B7BE 6609D2                 <1>     or     dx, dx ; If bytes per cluster > 32KB it is invalid
0000B7C1 755D                   <1>     jnz    short loc_lcde_invalid_format
                                <1>     ;mov   ecx, eax
0000B7C3 6689C1                 <1>     mov    cx, ax ; BYTES PER CLUSTER
0000B7C6 A1[488B0100]           <1>     mov    eax, [LCDE_ByteOffset]
                                <1>     ;sub   edx, edx
0000B7CB F7F1                   <1>     div    ecx
0000B7CD 3DFFFF0000             <1>     cmp    eax, 65535
0000B7D2 774C                   <1>     ja     short loc_lcde_invalid_format
                                <1>
                                <1>     ; cluster sequence number of directory (< 65536)
0000B7D4 66A3[428B0100]         <1>     mov    [LCDE_ClusterSN], ax
                                <1>
0000B7DA 6689D0                 <1>     mov    ax, dx ; byte offset in cluster (directory buffer)
0000B7DD 66BB2000               <1>     mov    bx, 32 ; ; 1 dir entry = 32 bytes
0000B7E1 6629D2                 <1>     sub    dx, dx  ; 0
0000B7E4 66F7F3                 <1>     div    bx
0000B7E7 66A3[408B0100]         <1>     mov    [LCDE_EntryIndex], ax ; dir entry index/sequence number
                                <1>                                  ; (in directory buffer/cluster)
                                <1> loc_lcde_get_current_sub_dir_fcluster:
0000B7ED A1[68810100]           <1>     mov    eax, [Current_Dir_FCluster]
                                <1>
```

```
                                <1> loc_lcde_get_next_cluster:
0000B7F2 66833D[428B0100]00     <1>     cmp     word [LCDE_ClusterSN], 0
0000B7FA 763E                   <1>     jna     short loc_lcde_check_dir_buffer_cluster
0000B7FC A3[448B0100]           <1>     mov     [LCDE_Cluster], eax
0000B801 E834100000             <1>     call    get_next_cluster
0000B806 7220                   <1>     jc      short loc_lcde_check_gnc_error
0000B808 66FF0D[428B0100]       <1>     dec     word [LCDE_ClusterSN]
0000B80F EBE1                   <1>     jmp     short loc_lcde_get_next_cluster
                                <1>
                                <1> loc_lcde_reload_current_directory:
0000B811 51                     <1>     push    ecx
0000B812 E813F6FFFF             <1>     call    reload_current_directory
0000B817 59                     <1>     pop     ecx
0000B818 0F8361FFFFFF           <1>     jnc     loc_lcde_cdl_check
0000B81E 5E                     <1>     pop     esi
0000B81F C3                     <1>     retn
                                <1>
                                <1> loc_lcde_invalid_format:
                                <1>     ; 15/10/2016 (0Bh -> 28)
0000B820 B81C000000             <1>     mov     eax, 28 ; Invalid Format !
                                <1> loc_lcde_drive_not_ready_read_err:
0000B825 F9                     <1>     stc
0000B826 5E                     <1>     pop     esi
0000B827 C3                     <1>     retn
                                <1>
                                <1> loc_lcde_check_gnc_error:
0000B828 09C0                   <1>     or      eax, eax
0000B82A 75F9                   <1>     jnz     short loc_lcde_drive_not_ready_read_err
0000B82C 66FF0D[428B0100]       <1>     dec     word [LCDE_ClusterSN]
0000B833 75EB                   <1>     jnz     short loc_lcde_invalid_format
0000B835 A1[448B0100]           <1>     mov     eax, [LCDE_Cluster]
                                <1>
                                <1> loc_lcde_check_dir_buffer_cluster:
0000B83A 3B05[99880100]         <1>     cmp     eax, [DirBuff_Cluster]
0000B840 755C                   <1>     jne     short loc_lcde_load_dir_cluster
0000B842 803D[94880100]00       <1>     cmp     byte [DirBuff_ValidData], 0
0000B849 7727                   <1>     ja      short lcde_check_dir_buffer_cluster_next
0000B84B 803D[6C810100]00       <1>     cmp     byte [Current_Dir_Level], 0
0000B852 775F                   <1>     ja      short loc_lcde_load_dir_cluster_0
                                <1>     ; 27/02/2016
                                <1>     ; TRDOS v1 has bug here for FAT32 fs !
0000B854 807E0303               <1>     cmp     byte [esi+LD_FATType], 3  ; FAT32
0000B858 7359                   <1>     jnb     short loc_lcde_load_dir_cluster_0
                                <1>     ;
0000B85A 0FB74E17               <1>     movzx   ecx, word [esi+LD_BPB+RootDirEnts]
0000B85E 6683C10F               <1>     add     cx, 15 ; round up (16 entries per sector)
0000B862 66C1E904               <1>     shr     cx, 4 ; 1 sector contains 16 dir entries
                                <1>
0000B866 8B4664                 <1>     mov     eax, [esi+LD_ROOTBegin]
0000B869 EB54                   <1>     jmp     short loc_lcde_load_dir_cluster_1
                                <1>
                                <1> loc_lcde_validate_dirBuff:
0000B86B C605[94880100]01       <1>     mov     byte [DirBuff_ValidData], 1
                                <1>
                                <1> lcde_check_dir_buffer_cluster_next:
0000B872 0FB71D[408B0100]       <1>     movzx   ebx, word [LCDE_EntryIndex]
0000B879 663B1D[97880100]       <1>     cmp     bx, [DirBuff_LastEntry]
0000B880 779E                   <1>     ja      short loc_lcde_invalid_format
0000B882 B820000000             <1>     mov     eax, 32
0000B887 F7E3                   <1>     mul     ebx
                                <1>     ;or     edx, edx
                                <1>     ;jnz    short loc_lcde_invalid_format
                                <1>
0000B889 BF00000800             <1>     mov     edi, Directory_Buffer
0000B88E 01C7                   <1>     add     edi, eax ; add entry offset to buffer address
                                <1>
                                <1> loc_lcde_dir_buffer_last_check:
0000B890 A1[99880100]           <1>     mov     eax, [DirBuff_Cluster]
0000B895 0FB60D[94880100]       <1>     movzx   ecx, byte [DirBuff_ValidData]
                                <1>
                                <1> loc_lcde_retn:
0000B89C 5E                     <1>     pop     esi
0000B89D C3                     <1>     retn
                                <1>
                                <1> loc_lcde_load_dir_cluster:
                                <1>     ;cmp    byte [DirBuff_ValidData], 2
                                <1>     ;jne    short loc_lcde_load_dir_cluster_n2
0000B89E 50                     <1>     push    eax
0000B89F E8E6FCFFFF             <1>     call    save_directory_buffer
0000B8A4 58                     <1>     pop     eax
0000B8A5 72F5                   <1>     jc      short loc_lcde_retn
                                <1>
                                <1> loc_lcde_load_dir_cluster_n2:
0000B8A7 C605[94880100]00       <1>     mov     byte [DirBuff_ValidData], 0
0000B8AE A3[99880100]           <1>     mov     [DirBuff_Cluster], eax
                                <1>
                                <1> loc_lcde_load_dir_cluster_0:
0000B8B3 83E802                 <1>     sub     eax, 2
0000B8B6 0FB64E13               <1>     movzx   ecx, byte [esi+LD_BPB+SecPerClust]
0000B8BA F7E1                   <1>     mul     ecx
0000B8BC 034668                 <1>     add     eax, [esi+LD_DATABegin]
                                <1>
                                <1> loc_lcde_load_dir_cluster_1:
0000B8BF BB00000800             <1>     mov     ebx, Directory_Buffer
                                <1>     ; ecx = sector count
0000B8C4 E8D26C0000             <1>     call    disk_read
0000B8C9 73A0                   <1>     jnc     short loc_lcde_validate_dirBuff
                                <1>
                                <1>     ; 15/10/2016
                                <1>     ; (Disk read error instead of drv not ready err)
0000B8CB B811000000             <1>     mov     eax, 17 ; Drive not ready or read error !
0000B8D0 EBCA                   <1>     jmp     short loc_lcde_retn
                                <1>
                                <1>
                                <1> remove_file:
```

```
                        <1>       ; 15/10/2016
                        <1>       ; 28/02/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>       ; 10/04/2011 (FILE.ASM, 'proc_delete_file')
                        <1>       ; 09/08/2010
                        <1>       ; INPUT ->
                        <1>       ;      EDI = Directory Buffer Entry Address
                        <1>       ;       CX = Directory Buffer Entry Counter/Index
                        <1>       ;       BL = Longname Entry Length
                        <1>       ;       BH = Logical DOS Drive Number
                        <1>
0000B8D2 29C0           <1>       sub    eax, eax
0000B8D4 88FC           <1>       mov    ah, bh
0000B8D6 BE00010900     <1>       mov    esi, Logical_DOSDisks
0000B8DB 01C6           <1>       add    esi, eax
                        <1>
0000B8DD 807E0301       <1>       cmp    byte [esi+LD_FATType], 1
0000B8E1 7312           <1>       jnb    short loc_del_fat_file
                        <1>
0000B8E3 807E04A1       <1>       cmp    byte [esi+LD_FSType], 0A1h
0000B8E7 7406           <1>       je     short loc_del_fs_file
                        <1>
                        <1> loc_del_file_invalid_format:
0000B8E9 30E4           <1>       xor    ah, ah
                        <1>       ; 15/10/2016 (0Bh -> 28)
0000B8EB B01C           <1>       mov    al, 28  ; Invalid Format
0000B8ED F9             <1>       stc
0000B8EE C3             <1>       retn
                        <1>
                        <1> loc_del_fs_file:
0000B8EF E83F0F0000     <1>       call   delete_fs_file
0000B8F4 C3             <1>       retn
                        <1>
                        <1> loc_del_fat_file:
0000B8F5 E808000000     <1>       call   delete_directory_entry
0000B8FA 7205           <1>       jc     short loc_del_file_err_retn
                        <1>
                        <1> loc_delfile_unlink_cluster_chain:
0000B8FC E863170000     <1>       call   truncate_cluster_chain
                        <1>       ;jc    short loc_del_file_err_retn
                        <1>
                        <1> loc_delfile_return:
                        <1> loc_del_file_err_retn:
0000B901 C3             <1>       retn
                        <1>
                        <1> delete_directory_entry:
                        <1>       ; 15/10/2016
                        <1>       ; 28/02/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>       ; 01/08/2011 (DIR.ASM, 'proc_delete_directory_entry')
                        <1>       ; 10/04/2011
                        <1>       ; INPUT ->
                        <1>       ;      ESI = Logical Dos Drive Descripton Table Address
                        <1>       ;      EDI = Directory Buffer Entry Address
                        <1>       ;       CX = Directory Buffer Entry Counter/Index
                        <1>       ;       BL = Longname Entry Length
                        <1>       ; OUTPUT ->
                        <1>       ;      ESI = Logical dos drive descripton table address
                        <1>       ;      EAX = First cluster to be truncated/unlinked
                        <1>       ;       CF = 1 -> Error code in EAX (AL)
                        <1>       ;       CF = 0 & BH <> 0 -> LMDT write error  (BH = 1)
                        <1>       ;       CF = 0 & BL <> 0 -> Long name delete error (BL = FFh)
                        <1>       ;
                        <1>       ;  (EDI, EBX, ECX register contents will be changed)
                        <1>
0000B902 881D[D68A0100] <1>       mov    [DelFile_LNEL], bl
0000B908 66890D[D48A0100] <1>     mov    [DelFile_EntryCounter], cx
                        <1>
0000B90F 668B4714       <1>       mov    ax, [edi+20] ; First Cluster High Word
0000B913 C1E010         <1>       shl    eax, 16
0000B916 668B471A       <1>       mov    ax, [edi+26] ; First Cluster Low Word
                        <1>
0000B91A A3[D08A0100]   <1>       mov    [DelFile_FCluster], eax
                        <1>
                        <1> loc_del_short_name:
0000B91F C607E5         <1>       mov    byte [edi], 0E5h  ; Deleted sign
                        <1>
0000B922 C605[94880100]02 <1>     mov    byte [DirBuff_ValidData], 2
0000B929 E85CFCFFFF     <1>       call   save_directory_buffer
0000B92E 723D           <1>       jc     short loc_delete_direntry_err_return
                        <1>
                        <1> loc_del_long_name:
0000B930 0FB615[D68A0100] <1>     movzx  edx, byte [DelFile_LNEL]
0000B937 08D2           <1>       or     dl, dl
0000B939 7416           <1>       jz     short loc_del_dir_entry_update_parent_dir_lm_date
                        <1>
0000B93B 8835[D68A0100] <1>       mov    byte [DelFile_LNEL], dh ; 0
                        <1>
0000B941 0FB705[D48A0100] <1>     movzx  eax,  word [DelFile_EntryCounter]
0000B948 29D0           <1>       sub    eax, edx
                        <1>       ;jnc   short loc_del_long_name_continue
0000B94A 7205           <1>       jc     short loc_del_dir_entry_update_parent_dir_lm_date
                        <1>
                        <1> ;loc_del_direntry_inv_data_return: ; 15/10/2016 (0Dh -> 29)
                        <1> ;     mov    eax, 29 ; 0Dh (TRDOS 8086) ; Invalid data
                        <1> ;     retn
                        <1>
                        <1> loc_del_long_name_continue:
                        <1>       ; AX = Directory Entry Number of the long name last entry
0000B94C E897FDFFFF     <1>       call   delete_longname
                        <1>       ;jc    short loc_delete_direntry_err_return
                        <1>
                        <1> loc_del_dir_entry_update_parent_dir_lm_date:
0000B951 801D[D68A0100]00 <1>     sbb    byte [DelFile_LNEL], 0 ; 0FFh if cf = 1
                        <1>
0000B958 E8C8FCFFFF     <1>       call   update_parent_dir_lmdt
0000B95D B700           <1>       mov    bh, 0
```

```
5653 0000B95F 80D700              <1>       adc    bh, 0
5654                              <1>
5655 0000B962 8A1D[D68A0100]      <1>       mov    bl, byte [DelFile_LNEL]
5656                              <1>
5657                              <1> loc_delete_direntry_return:
5658 0000B968 A1[D08A0100]        <1>       mov    eax, [DelFile_FCluster]
5659                              <1> loc_delete_direntry_err_return:
5660 0000B96D C3                  <1>       retn
5661                              <1>
5662                              <1> rename_directory_entry:
5663                              <1>       ; 13/11/2017
5664                              <1>       ; 15/10/2016
5665                              <1>       ; 06/03/2016 (TRDOS 386 = TRDOS v2.0)
5666                              <1>       ; 01/08/2011 (DIR.ASM, 'proc_rename_directory_entry')
5667                              <1>       ; 19/11/2010
5668                              <1>       ; INPUT -> (Current Directory)
5669                              <1>       ;      CX = Directory Entry Number
5670                              <1>       ;       EAX = First Cluster number of file or directory
5671                              <1>       ;       EBX = Longname Length (dir entry count) (< 256)
5672                              <1>       ;       ESI = New file (or directory) name (no path).
5673                              <1>       ;            (ASCIIZ string)
5674                              <1>       ; OUTPUT ->
5675                              <1>       ;       CF = 0 -> successfull
5676                              <1>       ;       CF = 1 -> error code in EAX (AL)
5677                              <1>       ;
5678                              <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
5679                              <1>
5680 0000B96E 803D[6D810100]00    <1>       cmp    byte [Current_FATType], 0
5681 0000B975 7706                <1>       ja     short loc_rename_directory_entry
5682                              <1>
5683 0000B977 E8B80E0000          <1>       call   rename_fs_file_or_directory
5684 0000B97C C3                  <1>       retn
5685                              <1>
5686                              <1> loc_rename_directory_entry:
5687 0000B97D 881D[D68A0100]      <1>       mov    [DelFile_LNEL], bl
5688 0000B983 66890D[D48A0100]    <1>       mov    [DelFile_EntryCounter], cx
5689 0000B98A A3[D08A0100]        <1>       mov    [DelFile_FCluster], eax
5690                              <1>
5691 0000B98F 0FB7C1              <1>       movzx  eax, cx
5692 0000B992 E8BBFDFFFF          <1>       call   locate_current_dir_entry
5693 0000B997 7308                <1>       jnc    short loc_rename_direntry_check_fcluster
5694                              <1>
5695                              <1> loc_rename_direntry_pop_retn:
5696 0000B999 C3                  <1>       retn
5697                              <1>
5698                              <1> loc_rename_direntry_pop_invd_retn:
5699 0000B99A F9                  <1>       stc
5700                              <1> loc_rename_direntry_invd_retn:
5701                              <1>       ; 15/10/2016 (0Dh -> 29)
5702 0000B99B B81D000000          <1>       mov    eax, 29 ; Invalid data
5703                              <1> loc_rename_retn:
5704 0000B9A0 C3                  <1>       retn
5705                              <1>
5706                              <1> loc_rename_direntry_check_fcluster:
5707 0000B9A1 668B5714            <1>       mov    dx, [edi+20] ; First Cluster HW
5708 0000B9A5 C1E210              <1>       shl    edx, 16 ; 13/11/2017
5709 0000B9A8 668B571A            <1>       mov    dx, [edi+26] ; First Cluster LW
5710 0000B9AC 3B15[D08A0100]      <1>       cmp    edx, [DelFile_FCluster]
5711 0000B9B2 75E6                <1>       jne    short loc_rename_direntry_pop_invd_retn
5712                              <1>       ; ESI = New file (or directory) name. (ASCIIZ string)
5713                              <1>       ; 06/03/2016
5714                              <1>       ; TRDOS v2 - NOTE: 'convert_file_name' procedure
5715                              <1>       ; has been modified for eliminating following situation.
5716                              <1>       ;
5717                              <1>       ; TRDOS v1 - NOTE: If file/dir name is more than 11 bytes
5718                              <1>       ; without a dot, attributes (edi+11) byte will be overwritten !
5719                              <1>       ; (Dot file name input must be proper for 11 byte dir entry
5720                              <1>       ;  type file name output.)
5721 0000B9B4 E8A2F6FFFF          <1>       call   convert_file_name
5722                              <1>
5723 0000B9B9 C605[94880100]02    <1>       mov    byte [DirBuff_ValidData], 2
5724 0000B9C0 E8C5FBFFFF          <1>       call   save_directory_buffer
5725 0000B9C5 72D9                <1>       jc     short loc_rename_retn
5726                              <1>
5727                              <1> loc_rename_direntry_del_ln:
5728 0000B9C7 0FB615[D68A0100]    <1>       movzx  edx, byte [DelFile_LNEL]
5729 0000B9CE 08D2                <1>       or     dl, dl
5730 0000B9D0 7410                <1>       jz     short loc_rename_direntry_update_parent_dir_lm_date
5731                              <1>
5732 0000B9D2 0FB705[D48A0100]    <1>       movzx  eax, word [DelFile_EntryCounter]
5733 0000B9D9 29D0                <1>       sub    eax, edx
5734 0000B9DB 72BE                <1>       jc     short loc_rename_direntry_invd_retn
5735                              <1>
5736                              <1> loc_rename_direntry_del_ln_continue:
5737                              <1>       ; EAX = Directory Entry Number of the long name last entry
5738 0000B9DD E806FDFFFF          <1>       call   delete_longname
5739                              <1>
5740                              <1> loc_rename_direntry_update_parent_dir_lm_date:
5741 0000B9E2 E83EFCFFFF          <1>       call   update_parent_dir_lmdt
5742 0000B9E7 31C0                <1>       xor    eax, eax
5743 0000B9E9 C3                  <1>       retn
5744                              <1>
5745                              <1> move_source_file_to_destination_file:
5746                              <1>       ; 15/10/2016
5747                              <1>       ; 11/03/2016
5748                              <1>       ; 10/03/2016 (TRDOS 386 = TRDOS v2.0)
5749                              <1>       ; 01/08/2011 (FILE.ASM)
5750                              <1>       ; 04/08/2010
5751                              <1>       ;
5752                              <1>       ;   Phase 1 -> Check destination file,
5753                              <1>       ;              'not found' is required
5754                              <1>       ;   Phase 2 -> Check source file
5755                              <1>       ;              'found' and proper attributes is required
5756                              <1>       ;   Phase 3 -> Make destination directory entry,
5757                              <1>       ;            add new dir cluster or section if it is required
```

```
                        <1>       ;   Phase 4 -> Delete source directory entry.
                        <1>       ;       cf = 1 causes to return before the phase 4.
                        <1>       ;    (source file protection against any possible errors)
                        <1>       ;
                        <1>       ; 08/05/2011 major modification
                        <1>       ;            -> destination file deleting is removed
                        <1>       ;            for msdos move/rename compatibility.
                        <1>       ;            (Access denied error will return if
                        <1>       ;            the destination file is found...)
                        <1>       ; INPUT ->
                        <1>       ;      ESI = Source File Pathname (Asciiz)
                        <1>       ;        EDI = Destination File Pathname (Asciiz)
                        <1>       ;        AL = 0 --> Interrupt (System call)
                        <1>       ;        AL > 0 --> Command Interpreter (Question)
                        <1>       ;        AL = 1 --> Question Phase
                        <1>       ;        AL = 2 --> Progress Phase
                        <1>       ; OUTPUT ->
                        <1>       ;       cf = 0 -> OK
                        <1>       ;        EAX = Destination directory first cluster
                        <1>       ;        ESI = Logical DOS drive description table
                        <1>       ;        EBX = Destination file structure offset
                        <1>       ;        CX = 0 (CX > 0 --> calculate free space error)
                        <1>       ;        cf = 1 -> Error code in EAX (AL)
                        <1>       ;
                        <1>       ;  (EDX, ECX, EBX, ESI, EDI will be changed)
                        <1>
0000B9EA 3C02           <1>       cmp    al, 2
0000B9EC 0F847F010000   <1>       je     msftdf_df2_check_directory
0000B9F2 A2[568C0100]   <1>       mov    [move_cmd_phase], al
                        <1>
                        <1> msftdf_parse_sf_path:
                        <1>       ; ESI = ASCIIZ pathname (Source)
0000B9F7 57             <1>       push   edi
0000B9F8 BF[548B0100]   <1>       mov    edi, SourceFile_Drv
0000B9FD E824F7FFFF     <1>       call   parse_path_name
0000BA02 5E             <1>       pop    esi
0000BA03 7211           <1>       jc     short msftdf_psf_retn
                        <1>
                        <1> msftdf_parse_df_path:
                        <1>       ; ESI = ASCIIZ pathname   (Destination)
0000BA05 BF[D48B0100]   <1>       mov    edi, DestinationFile_Drv
0000BA0A E817F7FFFF     <1>       call   parse_path_name
0000BA0F 7306           <1>       jnc    short msftdf_check_sf_drv
                        <1>
0000BA11 3C01           <1>       cmp    al, 1 ; File or directory name is not existing
0000BA13 7602           <1>       jna    short msftdf_check_sf_drv
                        <1>
                        <1> msftdf_stc_retn:
0000BA15 F9             <1>       stc
                        <1> msftdf_psf_retn:
0000BA16 C3             <1>       retn
                        <1>
                        <1> msftdf_check_sf_drv:
0000BA17 A0[548B0100]   <1>       mov    al, [SourceFile_Drv]
                        <1>
                        <1> msftdf_check_df_drv:
0000BA1C 8A15[D48B0100] <1>       mov    dl, [DestinationFile_Drv]
                        <1>
                        <1> msftdf_compare_sf_df_drv:
0000BA22 29DB           <1>       sub    ebx, ebx
0000BA24 8A3D[6E810100] <1>       mov    bh, [Current_Drv]
0000BA2A 38C2           <1>       cmp    dl, al
0000BA2C 7409           <1>       je     short msftdf_check_sf_df_drv_ok
                        <1>
                        <1> msftdf_not_same_drv:
                        <1>       ; DL = source file's drive number
0000BA2E 88C6           <1>       mov    dh, al ; destination file's drive number
                        <1>       ; 15/10/2016 (11h -> 21)
0000BA30 B815000000     <1>       mov    eax, 21 ; Not the same drive
0000BA35 F9             <1>       stc
0000BA36 C3             <1>       retn
                        <1>
                        <1> msftdf_check_sf_df_drv_ok:
0000BA37 8815[578C0100] <1>       mov    [msftdf_sf_df_drv], dl
                        <1>
0000BA3D 29C0           <1>       sub    eax, eax
0000BA3F 88D4           <1>       mov    ah, dl
0000BA41 0500010900     <1>       add    eax, Logical_DOSDisks
0000BA46 A3[588C0100]   <1>       mov    [msftdf_drv_offset], eax
                        <1>
0000BA4B 38FA           <1>       cmp    dl, bh ; byte [Current_Drv]
0000BA4D 7407           <1>       je     short msftdf_df_check_directory
                        <1>
                        <1> msftdf_change_drv:
0000BA4F E85EC1FFFF     <1>       call   change_current_drive
0000BA54 726D           <1>       jc     short msftdf_df_error_retn
                        <1>
                        <1> msftdf_check_destination_file:
                        <1> msftdf_df_check_directory:
0000BA56 BE[D58B0100]   <1>       mov    esi, DestinationFile_Directory
0000BA5B 803E20         <1>       cmp    byte [esi], 20h
0000BA5E 760F           <1>       jna    short msftdf_df_find_1
                        <1>
                        <1> msftdf_df_change_directory:
0000BA60 FE05[55390100] <1>       inc    byte [Restore_CDIR]
0000BA66 30E4           <1>       xor    ah, ah ; CD_COMMAND sign -> 0
0000BA68 E8A3F0FFFF     <1>       call   change_current_directory
0000BA6D 7254           <1>       jc     short msftdf_df_error_retn
                        <1>
                        <1> ;msftdf_df_change_prompt_dir_string:
                        <1> ;     call   change_prompt_dir_string
                        <1>
                        <1> msftdf_df_find_1:
0000BA6F BE[168C0100]   <1>       mov    esi, DestinationFile_Name
0000BA74 803E20         <1>       cmp    byte [esi], 20h
```

```
5863 0000BA77 7631              <1>       jna    short msftdf_df_copy_sf_name
5864                            <1>
5865                            <1> msftdf_df_find_2:
5866 0000BA79 6631C0            <1>       xor    ax, ax ; DestinationFile_AttributesMask -> any/zero
5867 0000BA7C E8D4D4FFFF        <1>       call   find_first_file
5868 0000BA81 0F838D000000      <1>       jnc    msftdf_permission_denied_retn
5869                            <1>
5870                            <1> msftdf_df_check_error_code:
5871                            <1>       ;cmp   eax, 2 ; File not found error
5872 0000BA87 3C02              <1>       cmp    al, 2
5873 0000BA89 7537              <1>       jne    short msftdf_df_stc_retn
5874                            <1>
5875                            <1> msftdf_df_check_fname:
5876                            <1>       ; 15/10/2016
5877 0000BA8B BE[168C0100]      <1>       mov    esi, DestinationFile_Name ; *
5878 0000BA90 E87ED8FFFF        <1>       call   check_filename
5879 0000BA95 7307              <1>       jnc    short msftdf_convert_df_direntry_name
5880                            <1>       ; invalid file name chars !
5881 0000BA97 B81A000000        <1>       mov    eax, ERR_INV_FILE_NAME  ; 26
5882 0000BA9C EB24              <1>       jmp    short msftdf_df_stc_retn
5883                            <1>
5884                            <1> msftdf_convert_df_direntry_name:
5885                            <1>       ; mov  esi, DestinationFile_Name ; *
5886 0000BA9E BF[268C0100]      <1>       mov    edi, DestinationFile_DirEntry
5887 0000BAA3 E8B3F5FFFF        <1>       call   convert_file_name
5888 0000BAA8 EB1A              <1>       jmp    short msftdf_restore_current_dir_1
5889                            <1>
5890                            <1> msftdf_df_copy_sf_name:
5891 0000BAAA 89F7              <1>       mov    edi, esi
5892 0000BAAC 57                <1>       push   edi
5893 0000BAAD BE[968B0100]      <1>       mov    esi, SourceFile_Name
5894 0000BAB2 B90C000000        <1>       mov    ecx, 12
5895                            <1> msftdf_df_copy_sf_name_loop:
5896 0000BAB7 AC                <1>       lodsb
5897 0000BAB8 AA                <1>       stosb
5898 0000BAB9 08C0              <1>       or     al, al
5899 0000BABB 7402              <1>       jz     short msftdf_df_copy_sf_name_ok
5900 0000BABD E2F8              <1>       loop   msftdf_df_copy_sf_name_loop
5901                            <1> msftdf_df_copy_sf_name_ok:
5902 0000BABF 5E                <1>       pop    esi
5903 0000BAC0 EBB7              <1>       jmp    short msftdf_df_find_2
5904                            <1>
5905                            <1> msftdf_df_stc_retn:
5906 0000BAC2 F9                <1>       stc
5907                            <1> msftdf_restore_cdir_failed:
5908                            <1> msftdf_df_error_retn:
5909 0000BAC3 C3                <1>       retn
5910                            <1>
5911                            <1> msftdf_restore_current_dir_1:
5912 0000BAC4 803D[55390100]00  <1>       cmp    byte [Restore_CDIR], 0
5913 0000BACB 760D              <1>       jna    short msftdf_sf_check_directory
5914 0000BACD 8B35[588C0100]    <1>       mov    esi, [msftdf_drv_offset]
5915 0000BAD3 E891C1FFFF        <1>       call   restore_current_directory
5916 0000BAD8 72E9              <1>       jc     short msftdf_restore_cdir_failed
5917                            <1>
5918                            <1> msftdf_sf_check_directory:
5919 0000BADA BE[558B0100]      <1>       mov    esi, SourceFile_Directory
5920 0000BADF 803E20            <1>       cmp    byte [esi], 20h
5921 0000BAE2 760F              <1>       jna    short msftdf_sf_find
5922                            <1> msftdf_sf_change_directory:
5923 0000BAE4 FE05[55390100]    <1>       inc    byte [Restore_CDIR]
5924 0000BAEA 30E4              <1>       xor    ah, ah ; CD_COMMAND sign -> 0
5925 0000BAEC E81FF0FFFF        <1>       call   change_current_directory
5926 0000BAF1 7227              <1>       jc     short msftdf_return
5927                            <1>
5928                            <1> ;msftdf_sf_change_prompt_dir_string:
5929                            <1> ;     call   change_prompt_dir_string
5930                            <1>
5931                            <1> msftdf_sf_find:
5932 0000BAF3 BE[968B0100]      <1>       mov    esi, SourceFile_Name  ; Offset 66
5933 0000BAF8 66B80018          <1>       mov    ax, 1800h ; Only files
5934 0000BAFC E854D4FFFF        <1>       call   find_first_file
5935 0000BB01 7217              <1>       jc     short msftdf_return
5936                            <1>
5937                            <1> msftdf_sf_ambgfn_check:
5938 0000BB03 6609D2            <1>       or     dx, dx ; Ambiguous filename chars used sign (DX>0)
5939 0000BB06 7407              <1>       jz     short msftdf_sf_found
5940                            <1>
5941                            <1> msftdf_ambiguous_file_name_error:
5942 0000BB08 B802000000        <1>       mov    eax, 2 ; File not found error
5943 0000BB0D F9                <1>       stc
5944 0000BB0E C3                <1>       retn
5945                            <1>
5946                            <1> msftdf_sf_found:
5947 0000BB0F 80E31F            <1>       and    bl, 1Fh ; Attributes, D-V-S-H-R
5948 0000BB12 7416              <1>       jz     short msftdf_save_sf_structure
5949                            <1>
5950                            <1> msftdf_permission_denied_retn:
5951 0000BB14 B805000000        <1>       mov    eax, 05h ; Access (Permission) denied !
5952 0000BB19 F9                <1>       stc
5953                            <1> msftdf_rest_cdir_err_retn:
5954                            <1> msftdf_return:
5955 0000BB1A C3                <1>       retn
5956                            <1>
5957                            <1> msftdf_phase_1_return:
5958 0000BB1B 31C0              <1>       xor    eax, eax
5959 0000BB1D A2[568C0100]      <1>       mov    [move_cmd_phase], al ; 0
5960 0000BB22 FEC0              <1>       inc    al ; mov al, 1
5961 0000BB24 BB[71BB0000]      <1>       mov    ebx, msftdf_df2_check_directory
5962                            <1>       ;mov   edx, 0FFFFFFFFh
5963 0000BB29 C3                <1>       retn
5964                            <1>
5965                            <1> msftdf_save_sf_structure:
5966 0000BB2A BE[608A0100]      <1>       mov    esi, FindFile_DirEntry
5967 0000BB2F BF[A68B0100]      <1>       mov    edi, SourceFile_DirEntry
```

```
5968 0000BB34 B908000000          <1>       mov   ecx, 8
5969 0000BB39 F3A5                <1>       rep   movsd
5970                              <1>
5971                              <1> msftdf_df_copy_sf_parameters:
5972 0000BB3B BE0B000000          <1>       mov   esi, 11
5973 0000BB40 89F7                <1>       mov   edi, esi
5974 0000BB42 81C6[A68B0100]      <1>       add   esi, SourceFile_DirEntry
5975 0000BB48 81C7[268C0100]      <1>       add   edi, DestinationFile_DirEntry
5976                              <1>       ;mov  ecx, 21
5977 0000BB4E B115                <1>       mov   cl, 21
5978 0000BB50 F3A4                <1>       rep   movsb
5979                              <1>
5980                              <1> msftdf_restore_current_dir_2:
5981 0000BB52 803D[55390100]00    <1>       cmp   byte [Restore_CDIR], 0
5982 0000BB59 760D                <1>       jna   short msftdf_df2_check_move_cmd_phase
5983 0000BB5B 8B35[588C0100]      <1>       mov   esi, [msftdf_drv_offset]
5984 0000BB61 E803C1FFFF          <1>       call  restore_current_directory
5985 0000BB66 72B2                <1>       jc    short msftdf_rest_cdir_err_retn
5986                              <1>
5987                              <1> msftdf_df2_check_move_cmd_phase:
5988 0000BB68 803D[568C0100]01    <1>       cmp   byte [move_cmd_phase], 1
5989 0000BB6F 74AA                <1>       je    short msftdf_phase_1_return
5990                              <1>
5991                              <1> msftdf_df2_check_directory:
5992 0000BB71 BE[D58B0100]        <1>       mov   esi, DestinationFile_Directory
5993 0000BB76 803E20              <1>       cmp   byte [esi], 20h
5994 0000BB79 760F                <1>       jna   short msftdf_make_dfde_locate_ffe_on_directory
5995                              <1> msftdf_df2_change_directory:
5996 0000BB7B FE05[55390100]      <1>       inc   byte [Restore_CDIR]
5997 0000BB81 30E4                <1>       xor   ah, ah ; CD_COMMAND sign -> 0
5998 0000BB83 E888EFFFFF          <1>       call  change_current_directory
5999 0000BB88 7290                <1>       jc    short msftdf_return
6000                              <1>
6001                              <1> ;msftdf_df2_change_prompt_dir_string:
6002                              <1> ;     call  change_prompt_dir_string
6003                              <1>
6004                              <1> msftdf_make_dfde_locate_ffe_on_directory:
6005                              <1>       ; Current directory fcluster <> Directory buffer cluster
6006                              <1>       ; Current directory will be reloaded by
6007                              <1>       ; 'locate_current_dir_file' procedure
6008                              <1>       ;
6009                              <1>       ;xor   ax, ax
6010 0000BB8A 31C0                <1>       xor   eax, eax
6011 0000BB8C 89C1                <1>       mov   ecx, eax
6012 0000BB8E 6649                <1>       dec   cx ; FFFFh
6013                              <1>             ; CX = FFFFh -> find first deleted or free entry
6014                              <1>             ; ESI would be ASCIIZ filename address if the call
6015                              <1>             ; would not be for first free or deleted dir entry
6016 0000BB90 E8CEF1FFFF          <1>       call  locate_current_dir_file
6017 0000BB95 733F                <1>       jnc   msftdf_make_dfde_set_ff_dir_entry
6018                              <1>
6019                              <1>       ;cmp   eax, 2
6020 0000BB97 3C02                <1>         cmp  al, 2
6021 0000BB99 7537                <1>       jne   short msftdf_error_retn
6022                              <1>
6023                              <1> msftdf_add_new_dir_entry_check_fs:
6024 0000BB9B 8B35[588C0100]      <1>       mov   esi, [msftdf_drv_offset]
6025 0000BBA1 A1[99880100]        <1>       mov   eax, [DirBuff_Cluster]
6026 0000BBA6 807E0300            <1>       cmp   byte [esi+LD_FATType], 0
6027 0000BBAA 7711                <1>       ja    short msftdf_add_new_subdir_cluster
6028                              <1>
6029                              <1> msftdf_add_new_fs_subdir_section:
6030                              <1>       ;CL=0, CH=E5h --> deleted entry, CH=0 --> free entry
6031                              <1>         ;xor cx, cx
6032 0000BBAC 30ED                <1>       xor   ch, ch ; cx = 0 --> add a new subdir section
6033 0000BBAE E8830C0000          <1>       call  add_new_fs_section
6034 0000BBB3 721E                <1>         jc  short msftdf_dsfde_error_retn
6035                              <1>       ;mov  [createfile_LastDirCluster], eax
6036                              <1>
6037 0000BBB5 E8A30E0000          <1>       call  load_FS_sub_directory
6038                              <1>       ;mov  ebx, Directory_Buffer
6039 0000BBBA 7318                <1>       jnc   short msftdf_add_new_fs_subdir_section_ok
6040 0000BBBC C3                  <1>       retn
6041                              <1>
6042                              <1> msftdf_add_new_subdir_cluster:
6043 0000BBBD E881150000          <1>       call  add_new_cluster
6044 0000BBC2 720F                <1>       jc    short msftdf_dsfde_error_retn
6045                              <1>
6046                              <1>       ;mov  [createfile_LastDirCluster], eax
6047                              <1>
6048 0000BBC4 E8570E0000          <1>       call  load_FAT_sub_directory
6049 0000BBC9 7309                <1>       jnc   short msftdf_add_new_subdir_cluster_ok
6050                              <1>       ; EBX = Directory buffer address
6051                              <1>
6052                              <1> msftdf_ansdc_update_parent_dir_lmdt:
6053                              <1> msftdf_make_dfde_err_upd_pdir_lmdt:
6054 0000BBCB 50                  <1>       push  eax
6055 0000BBCC E854FAFFFF          <1>       call  update_parent_dir_lmdt
6056 0000BBD1 58                  <1>       pop   eax
6057                              <1>
6058                              <1> msftdf_error_retn:
6059 0000BBD2 F9                  <1>       stc
6060                              <1> msftdf_dsfde_restore_cdir_failed:
6061                              <1> msftdf_dsfde_error_retn:
6062 0000BBD3 C3                  <1>       retn
6063                              <1>
6064                              <1> msftdf_add_new_fs_subdir_section_ok:
6065                              <1> msftdf_add_new_subdir_cluster_ok:
6066 0000BBD4 89DF                <1>       mov   edi, ebx ; Directory buffer address
6067                              <1>
6068                              <1> msftdf_make_dfde_set_ff_dir_entry:
6069 0000BBD6 8B15[68810100]      <1>       mov   edx, [Current_Dir_FCluster]
6070 0000BBDC 8915[BC8C0100]      <1>       mov   [createfile_FFCluster], edx
6071                              <1>       ; EDI = Directory entry offset
6072 0000BBE2 BE[268C0100]        <1>       mov   esi, DestinationFile_DirEntry
```

```
6073 0000BBE7 B908000000          <1>       mov    ecx, 8
6074 0000BBEC F3A5                <1>       rep    movsd
6075                              <1>
6076 0000BBEE C605[94880100]02    <1>       mov    byte [DirBuff_ValidData], 2
6077 0000BBF5 E890F9FFFF          <1>       call   save_directory_buffer
6078 0000BBFA 72CF                <1>       jc     short msftdf_make_dfde_err_upd_pdir_lmdt
6079                              <1>
6080                              <1> msftdf_make_dfde_update_pdir_lmdt:
6081 0000BBFC E824FAFFFF          <1>       call   update_parent_dir_lmdt
6082                              <1>
6083                              <1> msftdf_dsfde_restore_current_dir_1:
6084 0000BC01 803D[55390100]00    <1>       cmp    byte [Restore_CDIR], 0
6085 0000BC08 760D                <1>       jna    short msftdf_dsfde_check_directory
6086 0000BC0A 8B35[588C0100]      <1>       mov    esi, [msftdf_drv_offset]
6087 0000BC10 E854C0FFFF          <1>       call   restore_current_directory
6088 0000BC15 72BC                <1>       jc     short msftdf_dsfde_restore_cdir_failed
6089                              <1>
6090                              <1> msftdf_dsfde_check_directory:
6091 0000BC17 BE[558B0100]        <1>       mov    esi, SourceFile_Directory
6092 0000BC1C 803E20              <1>       cmp    byte [esi], 20h
6093 0000BC1F 760F                <1>       jna    short msftdf_dsfde_find_file
6094                              <1>
6095                              <1> msftdf_dsfde_change_directory:
6096 0000BC21 FE05[55390100]      <1>       inc    byte [Restore_CDIR]
6097 0000BC27 28E4                <1>       sub    ah, ah ; CD_COMMAND sign -> 0
6098 0000BC29 E8E2EEFFFF          <1>       call   change_current_directory
6099 0000BC2E 72A3                <1>       jc     short msftdf_dsfde_error_retn
6100                              <1>
6101                              <1> ;msftdf_dsfde_sf_change_prompt_dir_string:
6102                              <1> ;     call   change_prompt_dir_string
6103                              <1>
6104                              <1> msftdf_dsfde_find_file:
6105 0000BC30 BE[968B0100]        <1>       mov    esi, SourceFile_Name  ; Offset 66
6106 0000BC35 668B460E            <1>       mov    ax, [esi+14] ; 80 -> SourceFile_AttributesMask
6107 0000BC39 E817D3FFFF          <1>       call   find_first_file
6108 0000BC3E 7293                <1>       jc     short msftdf_dsfde_error_retn
6109                              <1>
6110                              <1> msftdf_dsfde_delete_direntry:
6111 0000BC40 8B35[588C0100]      <1>       mov    esi, [msftdf_drv_offset]
6112                              <1>
6113 0000BC46 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
6114 0000BC4A 770A                <1>       ja     short msftdf_delete_FAT_direntry
6115                              <1>
6116 0000BC4C 30DB                <1>       xor    bl, bl
6117                              <1>       ; BL = 0 -> File
6118                              <1>       ; EDI -> Directory buffer entry offset/address
6119 0000BC4E E8E40B0000          <1>       call   delete_fs_directory_entry
6120 0000BC53 7315                <1>       jnc    short msftdf_dsfde_restore_current_dir_2
6121 0000BC55 C3                  <1>       retn
6122                              <1>
6123                              <1> msftdf_delete_FAT_direntry:
6124 0000BC56 8A1D[5D8A0100]      <1>       mov    bl, [FindFile_LongNameEntryLength]
6125 0000BC5C 668B0D[888A0100]    <1>       mov    cx, [FindFile_DirEntryNumber]
6126                              <1>       ; ESI = Logical DOS drive description table address
6127                              <1>       ; EDI = Directory buffer entry offset/address
6128 0000BC63 E89AFCFFFF          <1>       call   delete_directory_entry
6129 0000BC68 721C                <1>       jc     short msftdf_retn
6130                              <1>
6131                              <1> msftdf_dsfde_restore_current_dir_2:
6132 0000BC6A 803D[55390100]00    <1>       cmp    byte [Restore_CDIR], 0
6133 0000BC71 7607                <1>       jna    short msftdf_new_dir_fcluster_retn
6134                              <1>       ;mov   esi, [msftdf_drv_offset]
6135 0000BC73 E8F1BFFFFF          <1>       call   restore_current_directory
6136 0000BC78 720C                <1>       jc     short msftdf_retn
6137                              <1>
6138                              <1> msftdf_new_dir_fcluster_retn:
6139 0000BC7A 31C9                <1>       xor    ecx, ecx
6140 0000BC7C A1[BC8C0100]        <1>       mov    eax, [createfile_FFCluster]
6141 0000BC81 BB[D48B0100]        <1>       mov    ebx, DestinationFile_Drv
6142                              <1>
6143                              <1> msftdf_retn:
6144 0000BC86 C3                  <1>       retn
6145                              <1>
6146                              <1>
6147                              <1> copy_source_file_to_destination_file:
6148                              <1>       ; 17/10/2016
6149                              <1>       ; 16/10/2016
6150                              <1>       ; 15/10/2016
6151                              <1>       ; 30/03/2016, 31/03/2016
6152                              <1>       ; 24/03/2016, 25/03/2016, 28/03/2016
6153                              <1>       ; 21/03/2016, 22/03/2016, 23/03/2016
6154                              <1>       ; 16/03/2016, 17/03/2016, 18/03/2016
6155                              <1>       ; 15/03/2016 (TRDOS 386 = TRDOS v2.0)
6156                              <1>       ; 02/09/2011 (FILE.ASM 'copy_source_file_to_destination_file')
6157                              <1>       ; 01/08/2010 - 18/05/2011
6158                              <1>       ;
6159                              <1>       ;   Command Interpreter phase 1 enter ->
6160                              <1>       ;           AL = 1 -> Caller is command interpreter
6161                              <1>       ;           AL = 2 -> The second call, re-enter/continue
6162                              <1>       ;   Phase 1 -> Check source file
6163                              <1>       ;              'found' is required
6164                              <1>       ;   Phase 2 -> Check destination file,
6165                              <1>       ;              save 'found' or 'not found' status
6166                              <1>       ;              'permission denied' error will be return
6167                              <1>       ;              if attributes have not for ordinary file
6168                              <1>       ;              without readonly attribute
6169                              <1>       ;   Command Interpreter phase 1 return ->
6170                              <1>       ;              DH = Source file attributes
6171                              <1>       ;              DL = Destination file found status
6172                              <1>       ;              EAX = 0
6173                              <1>       ;   Command Interpreter phase 2 enter ->
6174                              <1>       ;              AL = 2 -> Continue from the last position
6175                              <1>       ;              AH =
6176                              <1>       ;   Phase 3 -> Load source file or use read/write cluster method
6177                              <1>       ;   Phase 4 -> Create destination file if it is not found
```

```
                                <1>       ;   Phase 5 -> Open destination file
                                <1>       ;   Phase 6 -> Read from source and write to destination
                                <1>       ;   Phase 7 -> Unload source file, if it is loaded at memory
                                <1>       ;       cf = 1 causes to return before the phase 7
                                <1>       ;             but loaded file will be unloaded
                                <1>       ;            (allocated memory block will be deallocated)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;      ESI = Source File Pathname (Asciiz)
                                <1>       ;       EDI = Destination File Pathname (Asciiz)
                                <1>       ;       AL = 0 --> Interrupt (System call)
                                <1>       ;       AL > 0 --> Command Interpreter (Question)
                                <1>       ;       AL = 1 --> Question Phase
                                <1>       ;       AL = 2 --> Progress Phase
                                <1>       ;
                                <1>       ; OUTPUT ->
                                <1>       ;     cf = 0 -> OK
                                <1>       ;     EAX = Destination file first cluster
                                <1>       ;
                                <1>       ;       CL > 0 if there is file reading error before EOF
                                <1>       ;            (incomplete copy)
                                <1>       ;       CH > 0 if file is (full) loaded at memory
                                <1>       ;
                                <1>       ;     cf = 1 -> Error code in AL (EAX)
                                <1>       ;
                                <1>       ; (EBX, ECX, ESI, EDI register contents will be changed)
                                <1>
                                <1>
0000BC87 3C02                   <1>       cmp    al, 2
0000BC89 0F845A020000           <1>       je     csftdf2_check_cdrv
                                <1>
                                <1> ; Phase 1
                                <1>
0000BC8F A2[7C8C0100]           <1>       mov    byte [copy_cmd_phase], al
                                <1>
0000BC94 57                     <1>       push   edi ; *
                                <1>
                                <1> csftdf_parse_sf_path:
0000BC95 BF[548B0100]           <1>       mov    edi, SourceFile_Drv
0000BC9A E887F4FFFF             <1>       call   parse_path_name
0000BC9F 721C                   <1>       jc     short csftdf_parse_sf_path_failed
                                <1>
                                <1> csftdf_parse_df_path:
0000BCA1 5E                     <1>       pop    esi ; * (pushed edi)
                                <1>
                                <1> csftdf_sf_check_filename_exists:
0000BCA2 803D[968B0100]21       <1>       cmp    byte [SourceFile_Name], 21h
0000BCA9 7215                   <1>       jb     short csftdf_sf_file_not_found_error
                                <1>
0000BCAB BF[D48B0100]           <1>       mov    edi, DestinationFile_Drv
0000BCB0 E871F4FFFF             <1>       call   parse_path_name
0000BCB5 7310                   <1>       jnc    short csftdf_check_sf_cdrv
                                <1>
0000BCB7 3C01                   <1>       cmp    al, 1 ; File or directory name is not existing
0000BCB9 760C                   <1>       jna    short csftdf_check_sf_cdrv
                                <1>
                                <1> csftdf_parse_df_path_failed:
0000BCBB F9                     <1>       stc
                                <1> csftdf_sf_error_retn:
0000BCBC C3                     <1>       retn
                                <1>
                                <1> csftdf_parse_sf_path_failed:
0000BCBD 5F                     <1>       pop    edi ; *
0000BCBE EBFC                   <1>       jmp    short csftdf_sf_error_retn
                                <1>
                                <1> csftdf_sf_file_not_found_error:
0000BCC0 B802000000             <1>       mov    eax, 2 ; File not found
0000BCC5 EBF5                   <1>       jmp    short csftdf_sf_error_retn
                                <1>
                                <1> csftdf_check_sf_cdrv:
0000BCC7 8A3D[6E810100]         <1>       mov    bh, [Current_Drv]
                                <1>
0000BCCD 883D[7F8C0100]         <1>       mov    [csftdf_cdrv], bh ; 23/03/2016
                                <1>
0000BCD3 8A15[548B0100]         <1>       mov    dl, [SourceFile_Drv]
0000BCD9 38FA                   <1>       cmp    dl, bh ; byte [Current_Drv]
0000BCDB 7407                   <1>       je     short csftdf_sf_check_directory
                                <1>
0000BCDD E8D0BEFFFF             <1>       call   change_current_drive
0000BCE2 72D8                   <1>       jc     short csftdf_sf_error_retn
                                <1>
                                <1> csftdf_sf_check_directory:
0000BCE4 BE[558B0100]           <1>       mov    esi, SourceFile_Directory
0000BCE9 803E20                 <1>       cmp    byte [esi], 20h
0000BCEC 760F                   <1>       jna    short csftdf_find_sf
                                <1>
                                <1> csftdf_sf_change_directory:
0000BCEE FE05[55390100]         <1>       inc    byte [Restore_CDIR]
0000BCF4 30E4                   <1>       xor    ah, ah ; CD_COMMAND sign -> 0
0000BCF6 E815EEFFFF             <1>       call   change_current_directory
0000BCFB 72BF                   <1>       jc     short csftdf_sf_error_retn
                                <1>
                                <1> ;csftdf_sf_change_prompt_dir_string:
                                <1> ;     call   change_prompt_dir_string
                                <1>
                                <1> csftdf_find_sf:
0000BCFD BE[968B0100]           <1>       mov    esi, SourceFile_Name
0000BD02 66B80018               <1>       mov    ax, 1800h ; Except volume label and dirs
0000BD06 E84AD2FFFF             <1>       call   find_first_file
0000BD0B 72AF                   <1>       jc     short csftdf_sf_error_retn
                                <1>
                                <1> csftdf_sf_ambgfn_check:
0000BD0D 6621D2                 <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
0000BD10 7407                   <1>       jz     short csftdf_sf_found
                                <1>
```

```
                                <1> csftdf_ambiguous_file_name_error:
0000BD12 B802000000             <1>     mov     eax, 2 ; File not found error
0000BD17 F9                     <1>     stc
0000BD18 C3                     <1>     retn
                                <1>
                                <1> csftdf_sf_found:
0000BD19 A3[808C0100]           <1>     mov     [csftdf_filesize], eax
                                <1>
0000BD1E 09C0                   <1>     or      eax, eax
0000BD20 7507                   <1>     jnz     short csftdf_set_source_file_direntry
                                <1>
                                <1> csftdf_sf_file_size_zero:
0000BD22 B814000000             <1>     mov     eax, 20 ; TRDOS zero length (file size) error
0000BD27 F9                     <1>     stc
0000BD28 C3                     <1>     retn
                                <1>
                                <1> csftdf_set_source_file_direntry:
0000BD29 BE[608A0100]           <1>     mov     esi, FindFile_DirEntry
0000BD2E BF[A68B0100]           <1>     mov     edi, SourceFile_DirEntry
0000BD33 B908000000             <1>     mov     ecx, 8
0000BD38 F3A5                   <1>     rep     movsd
                                <1>
                                <1> csftdf_sf_restore_cdrv:
                                <1>     ; 22/03/2016
0000BD3A 8A15[7F8C0100]         <1>     mov     dl, [csftdf_cdrv]
0000BD40 3A15[6E810100]         <1>     cmp     dl, [Current_Drv]
0000BD46 7407                   <1>     je      short csftdf_sf_restore_cdir
0000BD48 E865BEFFFF             <1>     call    change_current_drive
0000BD4D 724F                   <1>     jc      short csftdf_df_error_retn ; 30/03/2016
                                <1>
                                <1> csftdf_sf_restore_cdir:
0000BD4F 803D[55390100]00       <1>     cmp     byte [Restore_CDIR], 0
0000BD56 7612                   <1>     jna     short csftdf_df_check_filename_exists
0000BD58 29C0                   <1>     sub     eax, eax
0000BD5A BE00010900             <1>     mov     esi, Logical_DOSDisks
0000BD5F 88D4                   <1>     mov     ah, dl ; byte [csftdf_cdrv]
0000BD61 01C6                   <1>     add     esi, eax
0000BD63 E801BFFFFF             <1>     call    restore_current_directory
0000BD68 7234                   <1>     jc      short csftdf_df_error_retn
                                <1>
                                <1> csftdf_df_check_filename_exists:
0000BD6A 803D[168C0100]20       <1>     cmp     byte [DestinationFile_Name], 20h
0000BD71 7716                   <1>     ja      short csftdf_check_df_cdrv
                                <1>
                                <1> csftdf_copy_sf_name:
0000BD73 BF[168C0100]           <1>     mov     edi, DestinationFile_Name
0000BD78 BE[968B0100]           <1>     mov     esi, SourceFile_Name
0000BD7D B10C                   <1>     mov     cl, 12
                                <1>
                                <1> csftdf_df_copy_sf_name_loop:
0000BD7F AC                     <1>     lodsb
0000BD80 AA                     <1>     stosb
0000BD81 08C0                   <1>     or      al, al
0000BD83 7404                   <1>     jz      short csftdf_check_df_cdrv
0000BD85 FEC9                   <1>     dec     cl
0000BD87 75F6                   <1>     jnz     csftdf_df_copy_sf_name_loop
                                <1>
                                <1> csftdf_check_df_cdrv:
0000BD89 8A15[D48B0100]         <1>     mov     dl, [DestinationFile_Drv]
0000BD8F 3A15[6E810100]         <1>     cmp     dl, [Current_Drv]
0000BD95 7408                   <1>     je      short csftdf_df_check_directory
                                <1>
0000BD97 E816BEFFFF             <1>     call    change_current_drive
0000BD9C 7301                   <1>     jnc     short csftdf_df_check_directory
                                <1>
                                <1> csftdf_df_error_retn:
0000BD9E C3                     <1>     retn
                                <1>
                                <1> csftdf_df_check_directory:
0000BD9F BE[D58B0100]           <1>     mov     esi, DestinationFile_Directory
0000BDA4 803E20                 <1>     cmp     byte [esi], 20h
0000BDA7 760F                   <1>     jna     short csftdf_find_df
                                <1>
                                <1> csftdf_df_change_directory:
0000BDA9 FE05[55390100]         <1>     inc     byte [Restore_CDIR]
0000BDAF 28E4                   <1>     sub     ah, ah ; CD_COMMAND sign -> 0
0000BDB1 E85AEDFFFF             <1>     call    change_current_directory
0000BDB6 72E6                   <1>     jc      short csftdf_df_error_retn
                                <1>
                                <1> ;csftdf_df_change_prompt_dir_string:
                                <1> ;       call    change_prompt_dir_string
                                <1>
                                <1> csftdf_find_df:
                                <1>     ; 23/03/2016
0000BDB8 29DB                   <1>     sub     ebx, ebx
0000BDBA 8A3D[D48B0100]         <1>     mov     bh,  [DestinationFile_Drv]
0000BDC0 81C300010900           <1>     add     ebx, Logical_DOSDisks
0000BDC6 891D[AC8C0100]         <1>     mov     [csftdf_df_drv_dt], ebx
                                <1>
0000BDCC BE[168C0100]           <1>     mov     esi, DestinationFile_Name
0000BDD1 6631C0                 <1>     xor     ax, ax
                                <1>             ; DestinationFile_AttributesMask -> any/zero
0000BDD4 E87CD1FFFF             <1>     call    find_first_file
0000BDD9 7218                   <1>     jc      short csftdf_df_check_error_code
                                <1>
                                <1> csftdf_df_ambgfn_check:
0000BDDB 6609D2                 <1>     or      dx, dx ; Ambiguous filename chars used sign (DX>0)
0000BDDE 752A                   <1>     jnz     short csftdf_df_error_inv_fname
                                <1>
                                <1> csftdf_df_found:
0000BDE0 C605[7E8C0100]01       <1>     mov     byte [DestinationFileFound], 1
                                <1>     ; 17/10/2016 (cl -> bl)
0000BDE7 80E31F                 <1>     and     bl, 1Fh ; Attributes, D-V-S-H-R
0000BDEA 745F                   <1>     jz      short csftdf_df_save_first_cluster
                                <1>
```

```
                            <1> csftdf_df_permission_denied_retn:
0000BDEC B805000000          <1>       mov   eax, 05h ; Access/Permission denied.
                            <1> csftdf_df_error_stc_retn:
0000BDF1 F9                  <1>       stc
0000BDF2 C3                  <1>       retn
                            <1>
                            <1> csftdf_df_check_error_code:
                            <1>       ;cmp   eax, 2
0000BDF3 3C02                <1>       cmp   al, 2
0000BDF5 75FA                <1>       jne   short csftdf_df_error_stc_retn
                            <1>
0000BDF7 C605[7E8C0100]00    <1>       mov   byte [DestinationFileFound], 0
                            <1>
                            <1>       ; 15/10/2016
0000BDFE BE[508A0100]        <1>       mov   esi, FindFile_Name ; *
0000BE03 E80BD5FFFF          <1>       call  check_filename
0000BE08 7307                <1>       jnc   short csftdf_df_valid_fname
                            <1> csftdf_df_error_inv_fname: ; 'invalid file name !'
0000BE0A B81A000000          <1>       mov   eax, ERR_INV_FILE_NAME  ; 26
0000BE0F F9                  <1>       stc
0000BE10 C3                  <1>       retn
                            <1>
                            <1> csftdf_df_valid_fname:
                            <1>       ; 21/03/2016
                            <1>       ; (Capitalized file name)
                            <1>       ;mov   esi, FindFile_Name ; * ; 15/10/2016
0000BE11 BF[168C0100]        <1>       mov   edi, DestinationFile_Name
0000BE16 A5                  <1>       movsd
0000BE17 A5                  <1>       movsd
0000BE18 A5                  <1>       movsd
                            <1>       ;movsb
                            <1>
                            <1> csftdf_check_disk_free_size_0:
0000BE19 A1[C28B0100]        <1>       mov   eax, [SourceFile_DirEntry+DirEntry_FileSize]
                            <1>
                            <1> csftdf_check_disk_free_size_1:
                            <1>       ;sub   ebx, ebx
                            <1>       ;mov   esi, Logical_DOSDisks
                            <1>       ;mov   bh,  [DestinationFile_Drv]
                            <1>       ;add   esi, ebx
                            <1>
0000BE1E 8B35[AC8C0100]      <1>       mov   esi, [csftdf_df_drv_dt] ; 23/03/2016
                            <1>
0000BE24 0FB74E11            <1>       movzx ecx, word [esi+LD_BPB+BytesPerSec] ; 17, LD_BPB + 0Bh
0000BE28 01C8                <1>       add   eax, ecx
0000BE2A 48                  <1>       dec   eax  ; file size (additional bytes) + 511 (round up)
                            <1> csftdf_check_disk_free_size_3: ; 16/03/2016
0000BE2B 29D2                <1>       sub   edx, edx
0000BE2D F7F1                <1>       div   ecx ; bytes per sector
                            <1>
                            <1> csftdf_check_disk_free_size:
0000BE2F 3B4674              <1>       cmp   eax, [esi+LD_FreeSectors]
0000BE32 0F8294000000        <1>       jb    csftdf_check_disk_free_size_ok
0000BE38 770A                <1>       ja    short csftdf_df_insufficient_disk_space
                            <1>
0000BE3A 807E0300            <1>       cmp   byte [esi+LD_FATType], 0 ; FS needs FDT sector also.
0000BE3E 0F8788000000        <1>       ja    csftdf_check_disk_free_size_ok
                            <1>
                            <1> csftdf_df_insufficient_disk_space:
0000BE44 B827000000          <1>       mov   eax, 27h ; insufficient disk space
0000BE49 EBA6                <1>       jmp   short csftdf_df_error_stc_retn
                            <1>
                            <1> csftdf_df_save_first_cluster:
                            <1>       ; ESI = FindFile_DirEntry (for the old destination file)
                            <1>       ; EAX = Old destination file size
                            <1>       ; 24/03/2016
                            <1>       ; EDI = Directory entry address (within Dir Buffer boundaries)
0000BE4B 81EF00000800        <1>       sub   edi, Directory_Buffer  ; (<65536)
0000BE51 66C1EF05            <1>       shr   di, 5 ; Convert entry offset to entry index/number
0000BE55 66893D[4E8C0100]    <1>       mov   [DestinationFile_DirEntryNumber], di ; (<2048)
                            <1>
                            <1> csftdf_df_check_sf_df_fcluster:
0000BE5C 668B5614            <1>       mov   dx, [esi+DirEntry_FstClusHI]
0000BE60 C1E210              <1>       shl   edx, 16
0000BE63 668B561A            <1>       mov   dx, [esi+DirEntry_FstClusLO]
0000BE67 8915[908C0100]      <1>       mov   [csftdf_df_cluster], edx
                            <1> csftdf_df_check_sf_df_fcluster_1:
0000BE6D 668B15[BA8B0100]    <1>       mov   dx, [SourceFile_DirEntry+DirEntry_FstClusHI]
0000BE74 C1E210              <1>       shl   edx, 16
0000BE77 668B15[C08B0100]    <1>       mov   dx, [SourceFile_DirEntry+DirEntry_FstClusLO]
0000BE7E 3B15[908C0100]      <1>       cmp   edx, [csftdf_df_cluster]
0000BE84 7512                <1>       jne   short csftdf_df_check_sf_df_fcluster_ok
                            <1> csftdf_df_check_sf_df_drv:
0000BE86 8A15[548B0100]      <1>       mov   dl, [SourceFile_Drv]
0000BE8C 3A15[D48B0100]      <1>       cmp   dl, [DestinationFile_Drv]
0000BE92 7504                <1>       jne   short csftdf_df_check_sf_df_fcluster_ok
                            <1>
                            <1>       ; source and destination files are same !
                            <1>       ; (they have same first cluster value on same logical disk)
                            <1>
0000BE94 31C0                <1>       xor   eax, eax ; mov eax, 0 -> Bad command or file name !
0000BE96 F9                  <1>       stc
0000BE97 C3                  <1>       retn
                            <1>
                            <1> csftdf_df_check_sf_df_fcluster_ok:
                            <1> csftdf_df_move_findfile_struct:
                            <1>       ; mov  esi, FindFile_DirEntry
0000BE98 BF[268C0100]        <1>       mov   edi, DestinationFile_DirEntry
0000BE9D B908000000          <1>       mov   ecx, 8
0000BEA2 F3A5                <1>       rep   movsd
                            <1>
                            <1> csftdf_check_disk_free_size_2:
0000BEA4 89C2                <1>       mov   edx, eax ; Old destination file size
                            <1>
                            <1>       ;mov   eax, [SourceFile_DirEntry+DirEntry_FileSize]
```

```
6493 0000BEA6 A1[808C0100]        <1>       mov   eax, [csftdf_filesize] ; 23/03/2016
6494                              <1>
6495                              <1>       ;;sub  ecx, ecx ; 0
6496                              <1>       ;mov   esi, Logical_DOSDisks
6497                              <1>       ;mov   ch,  [DestinationFile_Drv]
6498                              <1>       ;add   esi, ecx
6499                              <1>       ;
6500                              <1>       ;mov   [csftdf_df_drv_dt], esi
6501                              <1>
6502 0000BEAB 8B35[AC8C0100]      <1>       mov   esi, [csftdf_df_drv_dt] ; 23/03/2016
6503                              <1>
6504 0000BEB1 668B4E11            <1>       mov   cx, [esi+LD_BPB+BytesPerSec] ; 17, LD_BPB + 0Bh
6505 0000BEB5 01CA                <1>       add   edx, ecx ; + 512
6506 0000BEB7 01C8                <1>       add   eax, ecx ; + 512
6507 0000BEB9 4A                  <1>       dec   edx ; old file size + 511 (round up)
6508 0000BEBA 48                  <1>       dec   eax ; new file size + 511 (round up)
6509 0000BEBB F7D9                <1>       neg   ecx ; -512 ; 0FFFFFE00h
6510 0000BEBD 21CA                <1>       and   edx, ecx ; = old sector count * 512
6511 0000BEBF 21C8                <1>       and   eax, ecx ; = new sector count * 512
6512                              <1>
6513 0000BEC1 29D0                <1>       sub   eax, edx ; new file size - old file size (on disk)
6514 0000BEC3 7607                <1>       jna   short csftdf_check_disk_free_size_ok
6515                              <1>
6516 0000BEC5 F7D9                <1>       neg   ecx ; 512 (bytes per sector) ; 200h
6517                              <1>       ; check free space for additional sectors
6518                              <1>       ; eax = number of additional sectors * bytes per sector
6519                              <1>       ; esi = Logical DOS drive number (of destination disk)
6520 0000BEC7 E95FFFFFFF          <1>         jmp     csftdf_check_disk_free_size_3
6521                              <1>
6522                              <1> csftdf_check_disk_free_size_ok:
6523                              <1>       ; 18/03/2016
6524                              <1> csftdf_df_check_copy_cmd_phase:
6525 0000BECC A0[7C8C0100]        <1>       mov   al, [copy_cmd_phase]
6526 0000BED1 3C01                <1>       cmp   al, 1
6527 0000BED3 7514                <1>       jne   short csftdf2_check_cdrv
6528                              <1>
6529 0000BED5 31C0                <1>       xor   eax, eax
6530 0000BED7 A2[7C8C0100]        <1>       mov   [copy_cmd_phase], al ; 0
6531                              <1>
6532 0000BEDC 8A15[7E8C0100]      <1>       mov   dl, [DestinationFileFound]
6533 0000BEE2 8A35[B18B0100]      <1>       mov   dh, [SourceFile_DirEntry+11] ; Attributes
6534                              <1>
6535                              <1> csftdf_return:
6536 0000BEE8 C3                  <1>       retn
6537                              <1>
6538                              <1> ; Phase 2
6539                              <1>
6540                              <1> csftdf2_check_cdrv:
6541                              <1>       ; 18/03/2016
6542                              <1>       ; Here, destination drive and directory are ready !
6543                              <1>       ; (checking/restoring is not needed)
6544                              <1>       ; (Since at the end of the phase 1)
6545                              <1>
6546                              <1> ;     mov   dl, [DestinationFile_Drv]
6547                              <1> ;     cmp   dl, [Current_Drv]
6548                              <1> ;     je    short csftdf2_df_check_directory
6549                              <1> ;
6550                              <1> ;     call  change_current_drive
6551                              <1> ;     jc    short csftdf2_read_error
6552                              <1> ;
6553                              <1> ;csftdf2_df_check_directory:
6554                              <1> ;     mov   esi, DestinationFile_Directory
6555                              <1> ;     cmp   byte [esi], 20h
6556                              <1> ;     jna   short csftdf2_df_check_found_or_not
6557                              <1> ;
6558                              <1> ;csftdf2_df_change_directory:
6559                              <1> ;     inc   byte [Restore_CDIR]
6560                              <1> ;     xor   ah, ah ; CD_COMMAND sign -> 0
6561                              <1> ;     call  change_current_directory
6562                              <1> ;     jc    short csftdf2_stc_return
6563                              <1> ;
6564                              <1> ;;csftdf2_df_change_prompt_dir_string:
6565                              <1> ;;    call  change_prompt_dir_string
6566                              <1>
6567                              <1> csftdf2_df_check_found_or_not:
6568                              <1>       ; 21/03/2016
6569 0000BEE9 803D[7E8C0100]00    <1>       cmp   byte [DestinationFileFound], 0
6570 0000BEF0 7739                <1>       ja    short csftdf2_set_sf_percentage
6571                              <1>
6572                              <1> csftdf2_create_file:
6573 0000BEF2 BE[168C0100]        <1>       mov   esi, DestinationFile_Name
6574 0000BEF7 A1[808C0100]        <1>       mov   eax, [csftdf_filesize]
6575 0000BEFC 30C9                <1>       xor   cl, cl ; 0
6576                              <1>
6577 0000BEFE 31DB                <1>       xor   ebx, ebx ; 0
6578 0000BF00 4B                  <1>       dec   ebx ; 0FFFFFFFFh
6579                              <1>
6580                              <1>       ; INPUT ->
6581                              <1>       ;     EAX -> File Size
6582                              <1>       ;     ESI = ASCIIZ File name
6583                              <1>       ;      CL = File attributes
6584                              <1>       ;     EBX = FFFFFFFFh -> empty file sign for FAT fs
6585                              <1>       ;     EBX <> FFFFFFFFh -> use file size for FAT fs
6586                              <1>       ;
6587                              <1>       ; OUTPUT ->
6588                              <1>       ;     EAX = New file's first cluster
6589                              <1>       ;     ESI = Logical Dos Drv Descr. Table Addr.
6590                              <1>       ;     EBX = CreateFile_Size address
6591                              <1>       ;     ECX = Sectors per cluster (<256)
6592                              <1>       ;     EDX = Directory Entry Index/Number (<65536)
6593                              <1>       ;
6594                              <1>       ;     cf = 1 -> error code in AL (EAX)
6595                              <1>
6596 0000BF01 E8EC050000          <1>       call  create_file
6597                              <1>       ;pop  esi
```

```
6598 0000BF06 0F82A3050000        <1>         jc      csftdf2_rw_error
6599                              <1>
6600                              <1> csftdf2_create_file_OK:
6601 0000BF0C A3[908C0100]        <1>         mov     [csftdf_df_cluster], eax
6602                              <1>
6603                              <1>         ; 24/03/2016
6604 0000BF11 668915[4E8C0100]    <1>         mov     [DestinationFile_DirEntryNumber], dx
6605                              <1>
6606                              <1>         ; 21/03/2016
6607 0000BF18 BE00000800          <1>         mov     esi, Directory_Buffer
6608 0000BF1D C1E205              <1>         shl     edx, 5 ; 32 * index number
6609 0000BF20 01D6                <1>         add     esi, edx
6610 0000BF22 BF[268C0100]        <1>         mov     edi, DestinationFile_DirEntry
6611 0000BF27 B108                <1>         mov     cl, 8 ; 32 bytes
6612 0000BF29 F3A5                <1>         rep     movsd
6613                              <1>
6614                              <1> csftdf2_set_sf_percentage:
6615                              <1>         ; 17/03/2016
6616 0000BF2B 31C0                <1>         xor     eax, eax
6617 0000BF2D A2[A48C0100]        <1>         mov     [csftdf_percentage], al ; 0, reset
6618                              <1>
6619 0000BF32 A3[9C8C0100]        <1>         mov     [csftdf_sf_rbytes], eax ; 0, reset
6620 0000BF37 A3[A08C0100]        <1>         mov     [csftdf_df_wbytes], eax ; 0, reset
6621                              <1>
6622 0000BF3C 8A25[548B0100]      <1>         mov     ah, [SourceFile_Drv]
6623 0000BF42 BE00010900          <1>         mov     esi, Logical_DOSDisks
6624 0000BF47 01C6                <1>         add     esi, eax
6625                              <1>
6626 0000BF49 8935[A88C0100]      <1>         mov     [csftdf_sf_drv_dt], esi ; 23/03/2016
6627                              <1>
6628 0000BF4F 668B15[BA8B0100]    <1>         mov     dx, [SourceFile_DirEntry+DirEntry_FstClusHI]
6629 0000BF56 C1E210              <1>         shl     edx, 16
6630 0000BF59 668B15[C08B0100]    <1>         mov     dx, [SourceFile_DirEntry+DirEntry_FstClusLO]
6631 0000BF60 8915[8C8C0100]      <1>         mov     [csftdf_sf_cluster], edx
6632                              <1>
6633                              <1>         ; 16/03/2016
6634                              <1>         ; Note: Singlix FS boot sector parameters (for cluster
6635                              <1>         ;       related calculations) has same offset
6636                              <1>         ;       values from LD_BPB as in FAT file system.
6637                              <1>         ;       [esi+LD_BPB+SecPerClust] is 1 for Singlix FS.
6638                              <1>         ;
6639 0000BF66 0FB64E13            <1>         movzx  ecx, byte [esi+LD_BPB+SecPerClust]
6640 0000BF6A 880D[D28B0100]      <1>         mov     [SourceFile_SecPerClust], cl
6641                              <1>
6642                              <1>         ; 17/03/2016
6643 0000BF70 386E03              <1>         cmp     [esi+LD_FATType], ch ; 0
6644 0000BF73 7707                <1>         ja      short csftdf2_set_sf_percent_rsize1
6645                              <1>
6646 0000BF75 B800000100          <1>         mov     eax, 65536 ; read/write buffer size for Singlix FS
6647 0000BF7A EB06                <1>         jmp     short csftdf2_set_sf_percent_rsize2
6648                              <1>
6649                              <1> csftdf2_set_sf_percent_rsize1:
6650 0000BF7C 668B4611            <1>         mov     ax, [esi+LD_BPB+BytesPerSec]
6651 0000BF80 F7E1                <1>         mul     ecx
6652                              <1>         ;sub    edx, edx
6653                              <1> csftdf2_set_sf_percent_rsize2:
6654 0000BF82 A3[948C0100]        <1>         mov     [csftdf_r_size], eax
6655                              <1>
6656                              <1> csftdf2_set_df_percentage:
6657                              <1>         ;sub    eax, eax
6658                              <1>         ;mov    ah, [DestinationFile_Drv]
6659                              <1>         ;mov    edi, Logical_DOSDisks
6660                              <1>         ;add    edi, eax
6661                              <1>         ;mov    [csftdf_df_drv_dt], edi ; 17/03/2016
6662                              <1>
6663 0000BF87 8B3D[AC8C0100]      <1>         mov     edi, [csftdf_df_drv_dt] ; 23/03/2016
6664                              <1>
6665                              <1>         ; 16/03/2016
6666                              <1>         ; Note: Singlix FS boot sector parameters (for cluster
6667                              <1>         ;       related calculations) has same offset
6668                              <1>         ;       values from LD_BPB as in FAT file system.
6669                              <1>         ;       [edi+LD_BPB+SecPerClust] is 1 for Singlix FS.
6670                              <1>         ;
6671                              <1>         ;movzx ecx, byte [edi+LD_BPB+SecPerClust]
6672 0000BF8D 8A4F13              <1>         mov     cl, [edi+LD_BPB+SecPerClust]
6673 0000BF90 880D[528C0100]      <1>         mov     [DestinationFile_SecPerClust], cl
6674                              <1>
6675                              <1>         ; 17/03/2016
6676 0000BF96 386F03              <1>         cmp     [edi+LD_FATType], ch ; 0
6677 0000BF99 7707                <1>         ja      short csftdf2_set_df_percent_wsize1
6678                              <1>
6679 0000BF9B B800000100          <1>         mov     eax, 65536 ; read/write buffer size for Singlix FS
6680 0000BFA0 EB06                <1>         jmp     short csftdf2_set_df_percent_wsize2
6681                              <1>
6682                              <1> csftdf2_set_df_percent_wsize1:
6683 0000BFA2 0FB74711            <1>         movzx  eax, word [edi+LD_BPB+BytesPerSec]
6684 0000BFA6 F7E1                <1>         mul     ecx
6685                              <1>         ;sub    edx, edx
6686                              <1> csftdf2_set_df_percent_wsize2:
6687 0000BFA8 A3[988C0100]        <1>         mov     [csftdf_w_size], eax
6688                              <1>
6689 0000BFAD A1[808C0100]        <1>         mov     eax, [csftdf_filesize]
6690                              <1>
6691 0000BFB2 3D00000100          <1>         cmp     eax, 65536 ; 64KB   ; small file
6692 0000BFB7 721F                <1>         jb      short csftdf2_load_file ; do not display percentage
6693                              <1>
6694                              <1> csftdf2_reset_wf_percent_ptr_chk_64k:
6695 0000BFB9 B201                <1>         mov     dl, 1 ; 25/03/2016
6696                              <1>
6697 0000BFBB 3D00000400          <1>         cmp     eax, 65536*4 ; 256KB
6698 0000BFC0 7310                <1>         jnb     short csftdf2_enable_percentage_display ; big file
6699                              <1>
6700                              <1>         ; 64-128KB file size for floppy disks
6701 0000BFC2 3815[548B0100]      <1>         cmp     byte [SourceFile_Drv], dl ; 1 ; read from floppy disk ?
6702 0000BFC8 7608                <1>         jna     short csftdf2_enable_percentage_display
```

```
6703                                 <1>
6704 0000BFCA 3815[D48B0100]        <1>       cmp   byte [DestinationFile_Drv], dl ; 1 ; write to floppy disk ?
6705 0000BFD0 7706                  <1>       ja    short csftdf2_load_file
6706                                 <1>
6707                                 <1> csftdf2_enable_percentage_display:
6708 0000BFD2 8815[A48C0100]        <1>       mov   [csftdf_percentage], dl ; 1
6709                                 <1>
6710                                 <1> csftdf2_load_file:
6711                                 <1>       ; 13/05/2016
6712                                 <1>       ; 19/03/2016
6713                                 <1>       ; 18/03/2016
6714                                 <1>       ; 17/03/2016
6715 0000BFD8 B40F                  <1>       mov   ah, 0Fh
6716 0000BFDA E81C57FFFF            <1>       call  _int10h
6717                                 <1>       ; 13/05/2016
6718 0000BFDF 883D[A58C0100]        <1>       mov   [csftdf_videopage], bh ; active video page
6719 0000BFE5 B403                  <1>       mov   ah, 03h
6720 0000BFE7 E80F57FFFF            <1>       call  _int10h
6721 0000BFEC 668915[A68C0100]      <1>       mov   [csftdf_cursorpos], dx
6722                                 <1>
6723 0000BFF3 29C0                  <1>       sub   eax, eax
6724 0000BFF5 A2[7D8C0100]          <1>       mov   [csftdf_rw_err], al ; 0
6725                                 <1>
6726                                 <1> ; ///
6727                                 <1> csftdf_sf_amb: ; 15/03/2016
6728 0000BFFA 8B0D[808C0100]        <1>       mov   ecx, [csftdf_filesize]    ; 23/03/2016
6729                                 <1>
6730                                 <1>       ; TRDOS 386 (TRDOS v2.0)
6731                                 <1>       ; Allocate contiguous memory block for loading the file
6732                                 <1>
6733                                 <1>       ;mov   ecx, [SourceFile_DirEntry+DirEntry_FileSize]
6734                                 <1>
6735                                 <1>       ;sub   eax, eax ; First free memory aperture
6736                                 <1>
6737                                 <1>       ; eax = 0 (Allocate memory from the beginning)
6738                                 <1>       ; ecx = File (Allocation) size in bytes
6739                                 <1>
6740 0000C000 E8929FFFFF            <1>       call  allocate_memory_block
6741 0000C005 7304                  <1>       jnc   short loc_check_sf_save_loading_parms
6742                                 <1>
6743 0000C007 29C0                  <1>       sub   eax, eax
6744 0000C009 29C9                  <1>       sub   ecx, ecx
6745                                 <1>
6746                                 <1> loc_check_sf_save_loading_parms:
6747 0000C00B A3[848C0100]          <1>       mov   [csftdf_sf_mem_addr], eax ; loading address
6748 0000C010 890D[888C0100]        <1>       mov   [csftdf_sf_mem_bsize], ecx ; block size
6749                                 <1> ; ///
6750                                 <1>       ; 19/03/2016
6751 0000C016 8B35[A88C0100]        <1>       mov   esi, [csftdf_sf_drv_dt] ; logical dos drv desc. tbl.
6752                                 <1>
6753                                 <1>       ; 17/03/2016
6754 0000C01C 09C0                  <1>       or    eax, eax ; contiguous free memory block address
6755 0000C01E 0F845B010000          <1>         jz      csftdf2_read_sf_cluster
6756                                 <1>
6757                                 <1>       ; 18/03/2016
6758 0000C024 8B1D[848C0100]        <1>       mov   ebx, [csftdf_sf_mem_addr] ; memory block address
6759                                 <1>
6760 0000C02A 807E0300              <1>       cmp   byte [esi+LD_FATType], 0
6761 0000C02E 0F8605020000          <1>         jna     csftdf2_load_fs_file
6762                                 <1>
6763                                 <1> csftdf2_load_fat_file:
6764 0000C034 53                    <1>       push  ebx ; *
6765                                 <1>
6766                                 <1> csftdf2_load_fat_file_next:
6767 0000C035 BE[A53F0100]          <1>       mov   esi, msg_reading
6768 0000C03A E806B0FFFF            <1>       call  print_msg
6769                                 <1>
6770 0000C03F 803D[A48C0100]00      <1>       cmp   byte [csftdf_percentage], 0
6771 0000C046 7605                  <1>       jna   short csftdf2_load_fat_file_1
6772                                 <1>
6773 0000C048 E87C000000            <1>       call  csftdf2_print_percentage ; 19/03/2016
6774                                 <1>
6775                                 <1> csftdf2_load_fat_file_1:
6776 0000C04D 8B35[A88C0100]        <1>       mov   esi, [csftdf_sf_drv_dt]
6777 0000C053 5B                    <1>       pop   ebx ; *
6778                                 <1>
6779                                 <1> csftdf2_load_fat_file_2:
6780 0000C054 E8B8000000            <1>       call  csftdf2_read_fat_file_sectors ; 19/03/2016
6781 0000C059 0F8250040000          <1>         jc      csftdf2_rw_error ; eocc! or disk error!
6782                                 <1>
6783 0000C05F 09D2                  <1>       or    edx, edx ; edx > 0 -> EOF
6784 0000C061 7520                  <1>       jnz   short csftdf2_load_fat_file_ok
6785                                 <1>
6786 0000C063 803D[A48C0100]00      <1>       cmp   byte [csftdf_percentage], 0
6787 0000C06A 76E8                  <1>       jna   short csftdf2_load_fat_file_2
6788                                 <1>
6789 0000C06C 53                    <1>       push  ebx ; *
6790                                 <1>
6791                                 <1>       ; Set cursor position
6792                                 <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
6793 0000C06D 8A3D[A58C0100]        <1>       mov   bh, [csftdf_videopage]
6794 0000C073 668B15[A68C0100]      <1>       mov   dx, [csftdf_cursorpos]
6795 0000C07A B402                  <1>       mov   ah, 2
6796 0000C07C E87A56FFFF            <1>       call  _int10h
6797 0000C081 EBB2                  <1>       jmp   short csftdf2_load_fat_file_next
6798                                 <1>
6799                                 <1> csftdf2_load_fat_file_ok:
6800 0000C083 803D[A48C0100]00      <1>       cmp   byte [csftdf_percentage], 0
6801 0000C08A 0F8651020000          <1>         jna     csftdf2_save_file ; 25/03/2016
6802                                 <1>
6803                                 <1>       ; "Reading... 100%"
6804 0000C090 BF[BD3F0100]          <1>       mov   edi, percentagestr
6805 0000C095 B031                  <1>       mov   al, '1'
6806 0000C097 AA                    <1>       stosb
6807 0000C098 B030                  <1>       mov   al, '0'
```

```
6808 0000C09A AA                <1>       stosb
6809 0000C09B AA                <1>       stosb
6810                            <1>
6811 0000C09C 8A3D[A58C0100]    <1>       mov   bh, [csftdf_videopage]
6812 0000C0A2 668B15[A68C0100]  <1>       mov   dx, [csftdf_cursorpos]
6813 0000C0A9 B402              <1>       mov   ah, 2
6814 0000C0AB E84B56FFFF        <1>       call  _int10h
6815                            <1>
6816 0000C0B0 BE[A53F0100]      <1>       mov   esi, msg_reading
6817 0000C0B5 E88BAFFFFF        <1>       call  print_msg
6818                            <1>
6819 0000C0BA BE[BD3F0100]      <1>       mov   esi, percentagestr
6820 0000C0BF E881AFFFFF        <1>       call  print_msg
6821                            <1>
6822 0000C0C4 E918020000        <1>         jmp     csftdf2_save_file ; 25/03/2016
6823                            <1>
6824                            <1> csftdf2_print_percentage:
6825                            <1>       ; 09/12/2017
6826                            <1>       ; 19/03/2016
6827                            <1>       ; 18/03/2016
6828 0000C0C9 B020              <1>       mov   al, 20h
6829 0000C0CB BF[BD3F0100]      <1>       mov   edi, percentagestr
6830 0000C0D0 AA                <1>       stosb
6831 0000C0D1 AA                <1>       stosb
6832 0000C0D2 A1[9C8C0100]      <1>       mov   eax, [csftdf_sf_rbytes]
6833 0000C0D7 BA64000000        <1>       mov   edx, 100
6834 0000C0DC F7E2              <1>       mul   edx
6835 0000C0DE 8B0D[808C0100]    <1>       mov   ecx, [csftdf_filesize]
6836 0000C0E4 F7F1              <1>       div   ecx
6837 0000C0E6 B10A              <1>       mov   cl, 10
6838 0000C0E8 F6F1              <1>       div   cl
6839 0000C0EA 80C430            <1>       add   ah, '0'
6840 0000C0ED 8827              <1>       mov   [edi], ah
6841 0000C0EF 20C0              <1>       and   al, al
6842 0000C0F1 740A              <1>       jz    short csftdf2_print_percent_1
6843 0000C0F3 4F                <1>       dec   edi
6844                            <1>       ;cbw
6845 0000C0F4 28E4              <1>       sub   ah, ah ; 09/12/2017
6846 0000C0F6 F6F1              <1>       div   cl
6847 0000C0F8 80C430            <1>       add   ah, '0'
6848 0000C0FB 8827              <1>       mov   [edi], ah
6849                            <1>       ;and  al, al
6850                            <1>       ;jz   short csftdf2_print_percent_1
6851                            <1>       ;dec  edi
6852                            <1>       ;mov  [edi], '1' ; 100%
6853                            <1>
6854                            <1> csftdf2_print_percent_1:
6855 0000C0FD BE[BD3F0100]      <1>       mov   esi, percentagestr
6856                            <1>       ;call print_msg
6857                            <1>       ;retn
6858 0000C102 E93EAFFFFF        <1>       jmp   print_msg
6859                            <1>
6860                            <1> csftdf2_read_file_sectors:
6861                            <1>       ; 19/03/2016
6862 0000C107 807E0300          <1>       cmp   byte [esi+LD_FATType], 0
6863 0000C10B 0F8627070000      <1>         jna     csftdf2_read_fs_file_sectors
6864                            <1>
6865                            <1> csftdf2_read_fat_file_sectors:
6866                            <1>       ; 19/03/2016
6867                            <1>       ; 18/03/2016
6868                            <1>       ; return:
6869                            <1>       ;   CF = 0 & EDX > 0 -> END OF FILE
6870                            <1>       ;   CF = 0 & EDX = 0 -> not EOF
6871                            <1>       ;   CF = 1 -> read error (error code in AL)
6872                            <1>
6873                            <1> csftdf2_read_fat_file_secs_0:
6874 0000C111 8B15[808C0100]    <1>       mov   edx, [csftdf_filesize]
6875 0000C117 2B15[9C8C0100]    <1>       sub   edx, [csftdf_sf_rbytes]
6876 0000C11D 3B15[948C0100]    <1>       cmp   edx, [csftdf_r_size]
6877 0000C123 7306              <1>       jnb   short csftdf2_read_fat_file_secs_1
6878 0000C125 8915[948C0100]    <1>       mov   [csftdf_r_size], edx
6879                            <1>
6880                            <1> csftdf2_read_fat_file_secs_1:
6881 0000C12B A1[948C0100]      <1>       mov   eax, [csftdf_r_size]
6882 0000C130 29D2              <1>       sub   edx, edx
6883 0000C132 0FB74E11          <1>       movzx ecx, word [esi+LD_BPB+BytesPerSec]
6884 0000C136 01C8              <1>       add   eax, ecx
6885 0000C138 48                <1>       dec   eax
6886 0000C139 F7F1              <1>       div   ecx
6887 0000C13B 89C1              <1>       mov   ecx, eax ; sector count
6888 0000C13D A1[8C8C0100]      <1>       mov   eax, [csftdf_sf_cluster]
6889                            <1>
6890                            <1>       ; EBX = memory block address (current)
6891                            <1>
6892 0000C142 E821090000        <1>       call  read_fat_file_sectors
6893 0000C147 7235              <1>       jc    short csftdf2_read_fat_file_secs_3
6894                            <1>
6895                            <1>       ; EBX = next memory address
6896                            <1>
6897 0000C149 A1[9C8C0100]      <1>       mov   eax, [csftdf_sf_rbytes]
6898 0000C14E 0305[948C0100]    <1>       add   eax, [csftdf_r_size]
6899 0000C154 8B15[808C0100]    <1>       mov   edx, [csftdf_filesize]
6900 0000C15A 39D0              <1>       cmp   eax, edx
6901 0000C15C 7320              <1>       jnb   short csftdf2_read_fat_file_secs_3 ; edx > 0
6902 0000C15E A3[9C8C0100]      <1>       mov   [csftdf_sf_rbytes], eax
6903                            <1>
6904 0000C163 53                <1>       push  ebx ; *
6905                            <1>       ; get next cluster (csftdf_r_size! bytes)
6906 0000C164 A1[8C8C0100]      <1>       mov   eax, [csftdf_sf_cluster]
6907 0000C169 E8CC060000        <1>       call  get_next_cluster
6908 0000C16E 5B                <1>       pop   ebx ; *
6909 0000C16F 7306              <1>       jnc   short csftdf2_read_fat_file_secs_2
6910                            <1>
6911                            <1>       ; 15/10/2016
6912                            <1>       ;Disk read error instad of drv not ready err
```

```
6913 0000C171 B811000000          <1>       mov    eax, 17 ; Read error !
6914 0000C176 C3                  <1>       retn
6915                              <1>
6916                              <1> csftdf2_read_fat_file_secs_2:
6917 0000C177 29D2                <1>       sub    edx, edx ; 0
6918 0000C179 A3[8C8C0100]        <1>       mov    [csftdf_sf_cluster], eax ; next cluster
6919                              <1>
6920                              <1> csftdf2_read_fat_file_secs_3:
6921 0000C17E C3                  <1>       retn
6922                              <1>
6923                              <1> csftdf2_read_sf_cluster:
6924                              <1>       ; 19/03/2016
6925 0000C17F BB00000700          <1>       mov    ebx, Cluster_Buffer ; buffer address (64KB)
6926                              <1>
6927 0000C184 803D[A48C0100]00    <1>       cmp    byte [csftdf_percentage], 0
6928 0000C18B 760D                <1>       jna    short csftdf2_read_sf_clust_2
6929                              <1>
6930 0000C18D 53                  <1>       push   ebx ; *
6931                              <1>
6932                              <1> csftdf2_read_sf_clust_next:
6933 0000C18E E836FFFFFF          <1>       call   csftdf2_print_percentage
6934                              <1>
6935                              <1> csftdf2_read_sf_clust_0:
6936 0000C193 8B35[A88C0100]      <1>       mov    esi, [csftdf_sf_drv_dt]
6937                              <1> csftdf2_read_sf_clust_1:
6938 0000C199 5B                  <1>       pop    ebx ; *
6939                              <1>
6940                              <1> csftdf2_read_sf_clust_2:
6941 0000C19A 89DA                <1>       mov    edx, ebx
6942 0000C19C 0315[948C0100]      <1>       add    edx, [csftdf_r_size]
6943 0000C1A2 81FA00000800        <1>       cmp    edx, Cluster_Buffer + 65536
6944 0000C1A8 772F                <1>       ja     short csftdf2_write_df_cluster
6945                              <1>
6946 0000C1AA E858FFFFFF          <1>       call   csftdf2_read_file_sectors ; 19/03/2016
6947 0000C1AF 0F8280020000        <1>         jc     csftdf2_save_fat_file_err2 ; eocc! or disk error!
6948                              <1>
6949 0000C1B5 09D2                <1>       or     edx, edx ; edx > 0 -> EOF
6950 0000C1B7 7520                <1>       jnz    short csftdf2_write_df_cluster
6951                              <1>
6952 0000C1B9 803D[A48C0100]00    <1>       cmp    byte [csftdf_percentage], 0
6953 0000C1C0 76D8                <1>       jna    short csftdf2_read_sf_clust_2
6954                              <1>
6955 0000C1C2 53                  <1>       push   ebx ; *
6956                              <1>
6957                              <1>       ; Set cursor position
6958                              <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
6959 0000C1C3 8A3D[A58C0100]      <1>       mov    bh, [csftdf_videopage]
6960 0000C1C9 668B15[A68C0100]    <1>       mov    dx, [csftdf_cursorpos]
6961 0000C1D0 B402                <1>       mov    ah, 2
6962 0000C1D2 E82455FFFF          <1>       call   _int10h
6963 0000C1D7 EBB5                <1>       jmp    short csftdf2_read_sf_clust_next
6964                              <1>
6965                              <1> csftdf2_write_df_cluster:
6966                              <1>       ; 19/03/2016
6967 0000C1D9 8B35[AC8C0100]      <1>       mov    esi, [csftdf_df_drv_dt]
6968 0000C1DF BB00000700          <1>       mov    ebx, Cluster_Buffer ; buffer address (64KB)
6969                              <1>
6970                              <1> csftdf2_write_df_clust_next:
6971 0000C1E4 E855000000          <1>       call   csftdf2_write_file_sectors ; 19/03/2016
6972 0000C1E9 0F8246020000        <1>       jc     csftdf2_save_fat_file_err2 ; eocc! or disk error!
6973                              <1>
6974 0000C1EF 09D2                <1>       or     edx, edx ; edx > 0 -> EOF
6975 0000C1F1 750A                <1>       jnz    short csftdf2_rw_f_clust_ok
6976                              <1>
6977 0000C1F3 81FB00000800        <1>       cmp    ebx, Cluster_Buffer + 65536
6978 0000C1F9 72E9                <1>       jb     short csftdf2_write_df_clust_next
6979                              <1>
6980 0000C1FB EB82                <1>       jmp    short csftdf2_read_sf_cluster
6981                              <1>
6982                              <1> csftdf2_rw_f_clust_ok:
6983 0000C1FD 803D[A48C0100]00    <1>       cmp    byte [csftdf_percentage], 0
6984 0000C204 0F86B2010000        <1>       jna    csftdf2_save_fat_file_4 ; 25/03/2016
6985                              <1>
6986                              <1>       ; "100%"
6987 0000C20A BF[BD3F0100]        <1>       mov    edi, percentagestr
6988 0000C20F B031                <1>       mov    al, '1'
6989 0000C211 AA                  <1>       stosb
6990 0000C212 B030                <1>       mov    al, '0'
6991 0000C214 AA                  <1>       stosb
6992 0000C215 AA                  <1>       stosb
6993                              <1>
6994 0000C216 8A3D[A58C0100]      <1>       mov    bh, [csftdf_videopage]
6995 0000C21C 668B15[A68C0100]    <1>       mov    dx, [csftdf_cursorpos]
6996 0000C223 B402                <1>       mov    ah, 2
6997 0000C225 E8D154FFFF          <1>       call   _int10h
6998                              <1>
6999 0000C22A BE[BD3F0100]        <1>       mov    esi, percentagestr
7000 0000C22F E811AEFFFF          <1>       call   print_msg
7001                              <1>
7002 0000C234 E983010000          <1>       jmp    csftdf2_save_fat_file_4
7003                              <1>
7004                              <1> csftdf2_load_fs_file:
7005                              <1>       ; temporary - 18/03/2016
7006 0000C239 E96F020000          <1>       jmp    csftdf2_read_error
7007                              <1>
7008                              <1> csftdf2_write_file_sectors:
7009                              <1>       ; 19/03/2016
7010 0000C23E 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
7011 0000C242 0F86F1050000        <1>       jna    csftdf2_write_fs_file_sectors
7012                              <1>
7013                              <1> csftdf2_write_fat_file_sectors:
7014                              <1>       ; 19/03/2016
7015                              <1>       ; 18/03/2016
7016                              <1>       ; return:
7017                              <1>       ;   CF = 0 & EDX > 0 -> END OF FILE
```

```
7018                                    <1>       ;   CF = 0 & EDX = 0 -> not EOF
7019                                    <1>       ;   CF = 1 -> write error (error code in AL)
7020                                    <1>
7021                                    <1> csftdf2_write_fat_file_secs_0:
7022 0000C248 8B15[808C0100]            <1>       mov   edx, [csftdf_filesize]
7023 0000C24E 2B15[A08C0100]            <1>       sub   edx, [csftdf_df_wbytes]
7024 0000C254 3B15[988C0100]            <1>       cmp   edx, [csftdf_w_size]
7025 0000C25A 7306                      <1>       jnb   short csftdf2_write_fat_file_secs_1
7026 0000C25C 8915[988C0100]            <1>       mov   [csftdf_w_size], edx
7027                                    <1>
7028                                    <1> csftdf2_write_fat_file_secs_1:
7029 0000C262 A1[988C0100]              <1>       mov   eax, [csftdf_w_size]
7030 0000C267 29D2                      <1>       sub   edx, edx
7031 0000C269 0FB74E11                  <1>       movzx ecx, word [esi+LD_BPB+BytesPerSec]
7032 0000C26D 01C8                      <1>       add   eax, ecx
7033 0000C26F 48                        <1>       dec   eax
7034 0000C270 F7F1                      <1>       div   ecx
7035 0000C272 89C1                      <1>       mov   ecx, eax ; sector count
7036 0000C274 A1[908C0100]              <1>       mov   eax, [csftdf_df_cluster]
7037                                    <1>
7038                                    <1>       ; EBX = memory block address (current)
7039                                    <1>
7040 0000C279 E8A20F0000                <1>       call  write_fat_file_sectors
7041 0000C27E 7259                      <1>       jc    short csftdf2_write_fat_file_secs_4
7042                                    <1>
7043                                    <1>       ; EBX = next memory address
7044                                    <1>
7045 0000C280 A1[A08C0100]              <1>       mov   eax, [csftdf_df_wbytes]
7046 0000C285 0305[988C0100]            <1>       add   eax, [csftdf_w_size]
7047 0000C28B 8B15[808C0100]            <1>       mov   edx, [csftdf_filesize]
7048 0000C291 39D0                      <1>       cmp   eax, edx
7049 0000C293 7344                      <1>       jnb   short csftdf2_write_fat_file_secs_4
7050 0000C295 A3[A08C0100]              <1>       mov   [csftdf_df_wbytes], eax
7051                                    <1>       ;
7052 0000C29A A3[428C0100]              <1>       mov   [DestinationFile_DirEntry+DirEntry_FileSize], eax
7053                                    <1>
7054 0000C29F 53                        <1>       push  ebx ; *
7055                                    <1>
7056 0000C2A0 803D[7E8C0100]01          <1>       cmp   byte [DestinationFileFound], 1
7057 0000C2A7 7210                      <1>       jb    short csftdf2_write_fat_file_secs_2
7058                                    <1>
7059                                    <1>       ; get next cluster (csftdf_w_size! bytes)
7060 0000C2A9 A1[908C0100]              <1>       mov   eax, [csftdf_df_cluster]
7061 0000C2AE E887050000                <1>       call  get_next_cluster
7062 0000C2B3 731C                      <1>       jnc   short csftdf2_write_fat_file_secs_3
7063                                    <1>
7064 0000C2B5 21C0                      <1>       and   eax, eax ; end of cluster chain!?
7065 0000C2B7 7521                      <1>       jnz   short csftdf2_write_fat_file_secs_5 ; disk error !
7066                                    <1>
7067                                    <1> csftdf2_write_fat_file_secs_2:
7068 0000C2B9 A1[908C0100]              <1>       mov   eax, [csftdf_df_cluster] ; last cluster
7069 0000C2BE E8800E0000                <1>       call  add_new_cluster
7070 0000C2C3 7215                      <1>       jc    short csftdf2_write_fat_file_secs_5
7071                                    <1>
7072                                    <1>       ; NOTE: Destination file size may be bigger than
7073                                    <1>       ; source file size when the last reading fails after here.
7074                                    <1>       ; (The last -empty- cluster of destination file must be
7075                                    <1>       ; truncated and LMDT must be current date&time for partial
7076                                    <1>       ; copy result!)
7077 0000C2C5 8B15[988C0100]            <1>       mov   edx, [csftdf_w_size] ; bytes per cluster
7078 0000C2CB 0115[428C0100]            <1>       add   [DestinationFile_DirEntry+DirEntry_FileSize], edx
7079                                    <1>
7080                                    <1> csftdf2_write_fat_file_secs_3:
7081 0000C2D1 5B                        <1>       pop   ebx ; *
7082 0000C2D2 29D2                      <1>       sub   edx, edx ; 0
7083 0000C2D4 A3[908C0100]              <1>       mov   [csftdf_df_cluster], eax ; next cluster
7084                                    <1>
7085                                    <1> csftdf2_write_fat_file_secs_4:
7086 0000C2D9 C3                        <1>       retn
7087                                    <1>
7088                                    <1> csftdf2_write_fat_file_secs_5:
7089 0000C2DA 5B                        <1>       pop   ebx ; *
7090                                    <1>       ; 16/10/2016 (1Dh -> 18)
7091 0000C2DB B812000000                <1>       mov   eax, 18 ; Write error !
7092 0000C2E0 C3                        <1>       retn
7093                                    <1>
7094                                    <1> csftdf2_save_file:
7095                                    <1>       ; 09/12/2017
7096                                    <1>       ; 25/03/2016
7097                                    <1>       ; 19/03/2016
7098                                    <1>       ; 18/03/2016
7099 0000C2E1 8B35[AC8C0100]            <1>       mov   esi, [csftdf_df_drv_dt] ; logical dos drv desc. tbl.
7100                                    <1>
7101 0000C2E7 8B1D[848C0100]            <1>       mov   ebx, [csftdf_sf_mem_addr] ; memory block address
7102                                    <1>
7103 0000C2ED 807E0300                  <1>       cmp   byte [esi+LD_FATType], 0
7104 0000C2F1 0F86F4010000              <1>       jna   csftdf2_save_fs_file
7105                                    <1>
7106                                    <1> csftdf2_save_fat_file:
7107 0000C2F7 53                        <1>       push  ebx; *
7108                                    <1>
7109 0000C2F8 803D[A48C0100]00          <1>       cmp   byte [csftdf_percentage], 0
7110 0000C2FF 7724                      <1>       ja    short csftdf2_save_fat_file_0
7111                                    <1>
7112                                    <1>       ; Set cursor position
7113                                    <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
7114 0000C301 8A3D[A58C0100]            <1>       mov   bh, [csftdf_videopage]
7115 0000C307 668B15[A68C0100]          <1>       mov   dx, [csftdf_cursorpos]
7116 0000C30E B402                      <1>       mov   ah, 2
7117 0000C310 E8E653FFFF                <1>       call  _int10h
7118                                    <1>
7119 0000C315 BE[B13F0100]              <1>       mov   esi, msg_writing
7120 0000C31A E826ADFFFF                <1>       call  print_msg
7121                                    <1>
7122                                    <1> csftdf2_save_fat_file_next:
```

```
7123 0000C31F 8B35[AC8C0100]      <1>       mov   esi, [csftdf_df_drv_dt] ; 25/03/2016
7124                              <1>
7125                              <1> csftdf2_save_fat_file_0:
7126 0000C325 5B                  <1>       pop   ebx ; *
7127                              <1>
7128                              <1> csftdf2_save_fat_file_1:
7129 0000C326 E813FFFFFF          <1>       call  csftdf2_write_file_sectors ; 19/03/2016
7130 0000C32B 0F827E010000        <1>       jc    csftdf2_rw_error ; eocc! or disk error!
7131                              <1>
7132 0000C331 09D2                <1>       or    edx, edx ; edx > 0 -> EOF
7133 0000C333 756D                <1>       jnz   short csftdf2_save_fat_file_3 ; 25/03/2016
7134                              <1>
7135 0000C335 803D[A48C0100]00    <1>       cmp   byte [csftdf_percentage], 0
7136 0000C33C 76E8                <1>       jna   short csftdf2_save_fat_file_1
7137                              <1>
7138 0000C33E B020                <1>       mov   al, 20h
7139 0000C340 BF[BD3F0100]        <1>       mov   edi, percentagestr
7140 0000C345 AA                  <1>       stosb
7141 0000C346 AA                  <1>       stosb
7142 0000C347 A1[A08C0100]        <1>       mov   eax, [csftdf_df_wbytes]
7143 0000C34C BA64000000          <1>       mov   edx, 100
7144 0000C351 F7E2                <1>       mul   edx
7145 0000C353 8B0D[808C0100]      <1>       mov   ecx, [csftdf_filesize]
7146 0000C359 F7F1                <1>       div   ecx
7147 0000C35B B10A                <1>       mov   cl, 10
7148 0000C35D F6F1                <1>       div   cl
7149 0000C35F 80C430              <1>       add   ah, '0'
7150 0000C362 8827                <1>       mov   [edi], ah
7151 0000C364 20C0                <1>       and   al, al
7152 0000C366 740A                <1>       jz    short csftdf2_save_fat_file_2
7153 0000C368 4F                  <1>       dec   edi
7154                              <1>       ;cbw
7155 0000C369 30E4                <1>       xor   ah, ah ; 09/12/2017
7156 0000C36B F6F1                <1>       div   cl
7157 0000C36D 80C430              <1>       add   ah, '0'
7158 0000C370 8827                <1>       mov   [edi], ah
7159                              <1>       ;and  al, al
7160                              <1>       ;jz   short csftdf2_save_fat_file_2
7161                              <1>       ;dec  edi
7162                              <1>       ;mov  [edi], '1' ; 100%
7163                              <1>
7164                              <1> csftdf2_save_fat_file_2:
7165 0000C372 53                  <1>       push  ebx ; *
7166                              <1>
7167 0000C373 E802000000          <1>       call  csftdf2_print_wr_percentage ; 25/03/2016
7168                              <1>
7169 0000C378 EBA5                <1>       jmp   csftdf2_save_fat_file_next
7170                              <1>
7171                              <1> csftdf2_print_wr_percentage:
7172                              <1>       ; Set cursor position
7173                              <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
7174 0000C37A 8A3D[A58C0100]      <1>       mov   bh, [csftdf_videopage]
7175 0000C380 668B15[A68C0100]    <1>       mov   dx, [csftdf_cursorpos]
7176 0000C387 B402                <1>       mov   ah, 2
7177 0000C389 E86D53FFFF          <1>       call  _int10h
7178                              <1>
7179 0000C38E BE[B13F0100]        <1>       mov   esi, msg_writing
7180 0000C393 E8ADACFFFF          <1>       call  print_msg
7181                              <1>
7182 0000C398 BE[BD3F0100]        <1>       mov   esi, percentagestr
7183                              <1>       ;call print_msg
7184                              <1>       ;retn
7185 0000C39D E9A3ACFFFF          <1>       jmp   print_msg
7186                              <1>
7187                              <1> csftdf2_save_fat_file_3:
7188 0000C3A2 803D[A48C0100]00    <1>       cmp   byte [csftdf_percentage], 0
7189 0000C3A9 7611                <1>       jna   csftdf2_save_fat_file_4 ; 25/03/2016
7190                              <1>
7191                              <1>       ; "100%"
7192 0000C3AB BF[BD3F0100]        <1>       mov   edi, percentagestr
7193 0000C3B0 B031                <1>       mov   al, '1'
7194 0000C3B2 AA                  <1>       stosb
7195 0000C3B3 B030                <1>       mov   al, '0'
7196 0000C3B5 AA                  <1>       stosb
7197 0000C3B6 AA                  <1>       stosb
7198                              <1>
7199 0000C3B7 E8BEFFFFFF          <1>       call  csftdf2_print_wr_percentage
7200                              <1>
7201                              <1> csftdf2_save_fat_file_4:
7202 0000C3BC 803D[7E8C0100]00    <1>       cmp   byte [DestinationFileFound], 0
7203 0000C3C3 7647                <1>       jna   short csftdf2_save_fat_file_6
7204                              <1>
7205 0000C3C5 8B35[AC8C0100]      <1>       mov   esi, [csftdf_df_drv_dt] ; 31/03/2016
7206                              <1>
7207 0000C3CB A1[908C0100]        <1>       mov   eax, [csftdf_df_cluster] ; last cluster
7208 0000C3D0 E865040000          <1>       call  get_next_cluster
7209 0000C3D5 7235                <1>       jc    short csftdf2_save_fat_file_6 ; eocc! or disk error!
7210                              <1>
7211 0000C3D7 A1[908C0100]        <1>       mov   eax, [csftdf_df_cluster] ; last cluster
7212                              <1>       ;xor  ecx, ecx
7213                              <1>       ;mov  [FAT_ClusterCounter], ecx ; 0 ; reset
7214                              <1>       ;dec  ecx ; 0FFFFFFFFh
7215                              <1>       ;shr  ecx, 4 ; 28 bit ; 0FFFFFFFh
7216 0000C3DC B9FFFFFF0F         <1>       mov   ecx, 0FFFFFFFh
7217 0000C3E1 E87E070000          <1>       call  update_cluster
7218 0000C3E6 7224                <1>       jc    short csftdf2_save_fat_file_6 ; really last cluster!?
7219                              <1>
7220 0000C3E8 A3[908C0100]        <1>       mov   [csftdf_df_cluster], eax ; next cluster
7221                              <1>
7222                              <1>       ; byte [FAT_BuffValidData] = 2
7223 0000C3ED E82F0A0000          <1>       call  save_fat_buffer
7224 0000C3F2 730E                <1>       jnc   short csftdf2_save_fat_file_5
7225                              <1>
7226 0000C3F4 8B15[808C0100]      <1>       mov   edx, [csftdf_filesize]
7227 0000C3FA 8915[428C0100]      <1>       mov   [DestinationFile_DirEntry+DirEntry_FileSize], edx
```

```
7228 0000C400 EB58                <1>       jmp    short csftdf2_save_fat_file_err3
7229                              <1>
7230                              <1> csftdf2_save_fat_file_5:
7231 0000C402 A1[908C0100]        <1>       mov    eax, [csftdf_df_cluster]
7232                              <1>
7233                              <1>       ; EAX = First cluster to be truncated/unlinked
7234                              <1>       ; ESI = Logical dos drive description table address
7235 0000C407 E8580C0000          <1>       call   truncate_cluster_chain
7236                              <1>
7237                              <1> csftdf2_save_fat_file_6:
7238                              <1>       ; 28/03/2016
7239 0000C40C BE[B18B0100]        <1>       mov    esi, SourceFile_DirEntry+DirEntry_Attr ; +11 to + 18
7240 0000C411 BF[318C0100]        <1>       mov    edi, DestinationFile_DirEntry+DirEntry_Attr ; +11 to + 18
7241 0000C416 A4                  <1>       movsb ; +11
7242 0000C417 A5                  <1>       movsd ; +12 .. +15
7243 0000C418 66A5                <1>       movsw ; +16 .. +17
7244                              <1>             ; + 18
7245 0000C41A 83C604              <1>       add    esi, 4
7246 0000C41D 83C704              <1>       add    edi, 4
7247 0000C420 A5                  <1>       movsd  ; DirEntry_WrtTime ; +22 .. +25
7248                              <1>
7249 0000C421 8B15[808C0100]      <1>       mov    edx, [csftdf_filesize]
7250 0000C427 8915[428C0100]      <1>       mov    [DestinationFile_DirEntry+DirEntry_FileSize], edx
7251                              <1>
7252 0000C42D E8BAF0FFFF          <1>       call   convert_current_date_time
7253                              <1>       ; DX = Date in dos dir entry format
7254                              <1>       ; AX = Time in dos dir entry format
7255 0000C432 EB4D                <1>       jmp    short csftdf2_save_fat_file_7
7256                              <1>
7257                              <1> csftdf2_save_fat_file_err1:
7258 0000C434 5B                  <1>       pop    ebx ; *
7259                              <1> csftdf2_save_fat_file_err2:
7260 0000C435 A1[A08C0100]        <1>       mov    eax, [csftdf_df_wbytes]
7261 0000C43A 8B15[428C0100]      <1>       mov    edx, [DestinationFile_DirEntry+DirEntry_FileSize]
7262 0000C440 39C2                <1>       cmp    edx, eax
7263 0000C442 7616                <1>       jna    short csftdf2_save_fat_file_err3
7264 0000C444 A1[908C0100]        <1>       mov    eax, [csftdf_df_cluster] ; last (empty) cluster
7265                              <1>       ; ESI = Logical dos drive description table address
7266 0000C449 E8160C0000          <1>       call   truncate_cluster_chain
7267 0000C44E 720A                <1>       jc     short csftdf2_save_fat_file_err3
7268 0000C450 A1[A08C0100]        <1>       mov    eax, [csftdf_df_wbytes]
7269 0000C455 A3[428C0100]        <1>       mov    [DestinationFile_DirEntry+DirEntry_FileSize], eax
7270                              <1> csftdf2_save_fat_file_err3:
7271 0000C45A E88DF0FFFF          <1>       call   convert_current_date_time
7272                              <1>       ; DX = Date in dos dir entry format
7273                              <1>       ; AX = Time in dos dir entry format
7274 0000C45F C605[338C0100]00    <1>       mov    byte [DestinationFile_DirEntry+DirEntry_CrtTimeTenth], 0
7275 0000C466 66A3[348C0100]      <1>       mov    [DestinationFile_DirEntry+DirEntry_CrtTime], ax
7276 0000C46C 668915[368C0100]    <1>       mov    [DestinationFile_DirEntry+DirEntry_CrtDate], dx
7277 0000C473 66A3[3C8C0100]      <1>       mov    [DestinationFile_DirEntry+DirEntry_WrtTime], ax
7278 0000C479 668915[3E8C0100]    <1>       mov    [DestinationFile_DirEntry+DirEntry_WrtDate], dx
7279 0000C480 F9                  <1>       stc
7280                              <1> csftdf2_save_fat_file_7:
7281 0000C481 9C                  <1>       pushf
7282 0000C482 668915[388C0100]    <1>       mov    [DestinationFile_DirEntry+DirEntry_LastAccDate], dx
7283 0000C489 BE[268C0100]        <1>       mov    esi, DestinationFile_DirEntry
7284 0000C48E BF00000800          <1>       mov    edi, Directory_Buffer
7285 0000C493 0FB70D[4E8C0100]    <1>       movzx  ecx, word [DestinationFile_DirEntryNumber] ; (<2048)
7286 0000C49A 66C1E105            <1>       shl    cx, 5 ; 32 * directory entry number
7287 0000C49E 01CF                <1>       add    edi, ecx
7288                              <1>       ;mov   ecx, 8
7289 0000C4A0 66B90800            <1>       mov    cx, 8
7290 0000C4A4 F3A5                <1>       rep    movsd
7291 0000C4A6 9D                  <1>       popf
7292 0000C4A7 730B                <1>       jnc    short csftdf2_write_file_OK
7293                              <1>
7294                              <1> csftdf2_write_error:
7295                              <1>       ; 18/03/2016
7296 0000C4A9 B01D                <1>       mov    al, 1Dh ; write error
7297 0000C4AB EB02                <1>       jmp    short csftdf2_rw_error
7298                              <1>
7299                              <1>       ; 16/03/2016
7300                              <1> csftdf2_read_error:
7301 0000C4AD B011                <1>       mov    al, 17 ; ; Drive not ready or read error!
7302                              <1> csftdf2_rw_error:
7303 0000C4AF A2[7D8C0100]        <1>       mov    [csftdf_rw_err], al
7304                              <1>
7305                              <1> csftdf2_write_file_OK:
7306                              <1>       ; 18/03/2016
7307 0000C4B4 C605[94880100]02    <1>       mov    byte [DirBuff_ValidData], 2
7308 0000C4BB E8CAF0FFFF          <1>       call   save_directory_buffer
7309                              <1>
7310                              <1>       ; Update last modification date&time of destination
7311                              <1>       ; file's (parent) directory
7312 0000C4C0 E860F1FFFF          <1>       call   update_parent_dir_lmdt
7313                              <1>       ;
7314 0000C4C5 A1[848C0100]        <1>       mov    eax, [csftdf_sf_mem_addr] ; start address
7315                              <1>
7316 0000C4CA 21C0                <1>       and    eax, eax
7317 0000C4CC 750E                <1>       jnz    short csftdf2_dealloc_mblock
7318                              <1>
7319 0000C4CE 88C5                <1>       mov    ch, al ; 0 (Cluster r/w, not full loading)
7320                              <1> csftdf2_dealloc_retn:
7321 0000C4D0 8A0D[7D8C0100]      <1>       mov    cl, [csftdf_rw_err]
7322 0000C4D6 A1[908C0100]        <1>       mov    eax, [csftdf_df_cluster]
7323 0000C4DB C3                  <1>       retn
7324                              <1>
7325                              <1> csftdf2_dealloc_mblock:
7326 0000C4DC 8B0D[888C0100]      <1>       mov    ecx, [csftdf_sf_mem_bsize] ; block size
7327 0000C4E2 E8BD9CFFFF          <1>       call   deallocate_memory_block
7328 0000C4E7 B5FF                <1>       mov    ch, 0FFh ; (File was full loaded at memory)
7329 0000C4E9 EBE5                <1>       jmp    short csftdf2_dealloc_retn
7330                              <1>
7331                              <1> csftdf2_save_fs_file:
7332                              <1>       ; 16/10/2016 (1Dh -> 18)
```

```
7333                                    <1>       ; temporary - (21/03/2016)
7334 0000C4EB B812000000                <1>       mov    eax, 18 ; write error
7335 0000C4F0 F9                        <1>       stc
7336 0000C4F1 C3                        <1>       retn
7337                                    <1>
7338                                    <1> create_file:
7339                                    <1>       ; 16/10/2016
7340                                    <1>       ; 24/03/2016, 31/03/2016
7341                                    <1>       ; 20/03/2016, 21/03/2016, 23/03/2016
7342                                    <1>       ; 19/03/2016 (TRDOS 396 = TRDOS v2.0)
7343                                    <1>       ; 03/09/2011 (FILE.ASM, 'proc_create_file')
7344                                    <1>       ; 09/08/2010
7345                                    <1>       ;
7346                                    <1>       ; INPUT ->
7347                                    <1>       ;      EAX = File Size
7348                                    <1>       ;      ESI = ASCIIZ File Name
7349                                    <1>       ;      CL = File Attributes
7350                                    <1>       ;      EBX = FFFFFFFFh -> create empty file
7351                                    <1>       ;                         (only for FAT fs)
7352                                    <1>       ; OUTPUT ->
7353                                    <1>       ;     CF = 0 ->
7354                                    <1>       ;      EAX = New file's first cluster
7355                                    <1>       ;      ESI = Logical Dos Drv Descr. Table Addr.
7356                                    <1>       ;      EBX = offset CreateFile_Size
7357                                    <1>       ;      ECX = Sectors per cluster (<256)
7358                                    <1>       ;      EDX = Directory entry index/number (<65536)
7359                                    <1>       ;     CF = 1 -> error code in AL
7360                                    <1>
7361                                    <1> ;     test   cl, 18h (directory or volume name)
7362                                    <1> ;     jnz    short loc_createfile_access_denied
7363 0000C4F2 80E107                    <1>       and    cl, 07h ; S, H, R
7364 0000C4F5 880D[CC8C0100]            <1>       mov    [createfile_attrib], cl
7365                                    <1>
7366 0000C4FB 89D9                      <1>       mov    ecx, ebx
7367 0000C4FD 89F3                      <1>       mov    ebx, esi ; ASCIIZ File Name address
7368 0000C4FF 29D2                      <1>       sub    edx, edx
7369 0000C501 8A35[6E810100]            <1>       mov    dh, [Current_Drv]
7370 0000C507 BE00010900                <1>       mov    esi, Logical_DOSDisks
7371 0000C50C 01D6                      <1>       add    esi, edx
7372                                    <1>
7373 0000C50E 8815[D78C0100]            <1>       mov    [createfile_UpdatePDir], dl ; 0 ; 31/03/2016
7374                                    <1>
7375                                    <1>       ; LD_DiskType = 0 for write protection (read only)
7376 0000C514 807E0101                  <1>       cmp    byte [esi+LD_DiskType], 1 ; 0 = Invalid
7377 0000C518 730A                      <1>       jnb    short loc_createfile_check_file_sytem
7378                                    <1>       ; 16/10/2016 (TRDOS Error code: 30, disk write protected)
7379 0000C51A B81E000000                <1>       mov    eax, 30 ; 13h, MSDOS err : Disk write-protected
7380 0000C51F 66BA0000                  <1>       mov    dx, 0
7381                                    <1>       ; err retn: EDX = 0, EBX = File name offset
7382                                    <1>       ; ESI -> Dos drive description table address
7383 0000C523 C3                        <1>       retn
7384                                    <1>
7385                                    <1> ;loc_createfile_access_denied:
7386                                    <1> ;     mov    eax, 05h ; access denied (invalid attributes input)
7387                                    <1> ;     stc
7388                                    <1> ;     retn
7389                                    <1>
7390                                    <1> loc_createfile_check_file_sytem:
7391 0000C524 807E0301                  <1>       cmp    byte [esi+LD_FATType], 1
7392 0000C528 730A                      <1>       jnb    short loc_createfile_chk_empty_FAT_file_sign1
7393                                    <1>
7394 0000C52A A3[B88C0100]              <1>       mov    [createfile_size], eax
7395                                    <1>       ; ESI = Logical Dos Drive Description Table address
7396                                    <1>       ; EBX = ASCIIZ File Name address
7397 0000C52F E9FE020000                <1>       jmp    create_fs_file
7398                                    <1>
7399                                    <1> loc_createfile_chk_empty_FAT_file_sign1:
7400                                    <1>       ; ECX = FFFFFFFFh -> create empty file if drive has FAT fs
7401 0000C534 41                        <1>       inc    ecx
7402 0000C535 7506                      <1>       jnz    short loc_createfile_chk_empty_FAT_file_sign2
7403 0000C537 890D[B88C0100]            <1>       mov    [createfile_size], ecx ; 0 ; empty file
7404                                    <1>
7405                                    <1> loc_createfile_chk_empty_FAT_file_sign2:
7406                                    <1>       ; 23/03/2016
7407 0000C53D 668B4E11                  <1>       mov    cx, [esi+LD_BPB+BytesPerSec]
7408 0000C541 66890D[D48C0100]          <1>       mov    [createfile_BytesPerSec], cx
7409                                    <1>
7410                                    <1>       ; EBX = ASCIIZ File Name address
7411 0000C548 0FB65613                  <1>       movzx  edx, byte [esi+LD_BPB+SecPerClust]
7412 0000C54C 8815[CD8C0100]            <1>       mov    [createfile_SecPerClust], dl
7413 0000C552 8B4E74                    <1>       mov    ecx, [esi+LD_FreeSectors]
7414 0000C555 39D1                      <1>       cmp    ecx, edx ; byte [createfile_SecPerClust]
7415 0000C557 7306                      <1>       jnb    short loc_create_fat_file
7416                                    <1>
7417                                    <1> loc_createfile_insufficient_disk_space:
7418 0000C559 B827000000                <1>       mov    eax, 27h
7419                                    <1> loc_createfile_gffc_retn:
7420 0000C55E C3                        <1>       retn
7421                                    <1>
7422                                    <1> loc_create_fat_file:
7423 0000C55F 891D[B08C0100]            <1>       mov    [createfile_Name_Offset], ebx
7424 0000C565 890D[B48C0100]            <1>       mov    [createfile_FreeSectors], ecx
7425                                    <1>
7426                                    <1> loc_createfile_gffc_1:
7427 0000C56B E821050000                <1>       call   get_first_free_cluster
7428 0000C570 72EC                      <1>       jc     short loc_createfile_gffc_retn
7429                                    <1>
7430 0000C572 A3[BC8C0100]              <1>       mov    [createfile_FFCluster], eax
7431                                    <1>
7432                                    <1> loc_createfile_locate_ffe_on_directory:
7433                                    <1>       ; Current directory fcluster <> Directory buffer cluster
7434                                    <1>       ; Current directory will be reloaded by
7435                                    <1>       ; 'locate_current_dir_file' procedure
7436                                    <1>       ;
7437                                    <1>       ; ESI = Logical Dos Drv Desc. Table Adress
```

```
7438 0000C577 56                <1>       push   esi ; *
7439 0000C578 31C0              <1>       xor    eax, eax
7440                            <1>
7441 0000C57A A3[8A880100]      <1>       mov    dword [FAT_ClusterCounter], eax ; 0
7442                            <1>       ; 21/03/2016
7443 0000C57F A2[D68C0100]      <1>       mov    byte [createfile_wfc], al ; 0
7444                            <1>
7445 0000C584 89C1              <1>       mov    ecx, eax
7446 0000C586 6649              <1>       dec    cx ; FFFFh
7447                            <1>       ; CX = FFFFh -> find first deleted or free entry
7448                            <1>       ; ESI would be ASCIIZ filename address if the call
7449                            <1>       ; would not be for first free or deleted dir entry
7450 0000C588 E8D6E7FFFF        <1>       call   locate_current_dir_file
7451 0000C58D 0F83EE000000      <1>       jnc    loc_createfile_set_ff_dir_entry
7452 0000C593 5E                <1>       pop    esi ; *
7453                            <1>        ; ESI = Logical DOS Drv. Description Table Address
7454 0000C594 83F802            <1>       cmp    eax, 2
7455 0000C597 7402              <1>       je     short loc_createfile_add_new_cluster
7456                            <1> loc_createfile_locate_file_stc_retn:
7457 0000C599 F9                <1>       stc
7458 0000C59A C3                <1>       retn
7459                            <1>
7460                            <1> loc_createfile_add_new_cluster:
7461 0000C59B 803D[6D810100]02  <1>       cmp    byte [Current_FATType], 2
7462                            <1>       ;cmp   byte [esi+LD_FATType], 2
7463 0000C5A2 770C              <1>       ja     short loc_createfile_add_new_cluster_check_fsc
7464 0000C5A4 803D[6C810100]01  <1>       cmp    byte [Current_Dir_Level], 1
7465                            <1>       ;cmp   byte [esi+LD_CDirLevel], 1
7466 0000C5AB 7303              <1>       jnb    short loc_createfile_add_new_cluster_check_fsc
7467                            <1>
7468                            <1>       ;mov   eax, 12
7469 0000C5AD B00C              <1>       mov    al, 12 ; No more files
7470                            <1>
7471                            <1> loc_createfile_anc_retn:
7472 0000C5AF C3                <1>       retn
7473                            <1>
7474                            <1> loc_createfile_add_new_cluster_check_fsc:
7475 0000C5B0 8B0D[B48C0100]    <1>       mov    ecx, [createfile_FreeSectors]
7476 0000C5B6 0FB605[CD8C0100]  <1>       movzx  eax, byte [createfile_SecPerClust]
7477 0000C5BD 66D1E0            <1>       shl    ax, 1 ; AX = 2 * AX
7478 0000C5C0 39C1              <1>       cmp    ecx, eax
7479 0000C5C2 7295              <1>       jb     short loc_createfile_insufficient_disk_space
7480                            <1>
7481                            <1> loc_createfile_add_new_subdir_cluster:
7482 0000C5C4 8B15[99880100]    <1>       mov    edx, [DirBuff_Cluster]
7483 0000C5CA 8915[C08C0100]    <1>       mov    [createfile_LastDirCluster], edx
7484                            <1>
7485 0000C5D0 A1[BC8C0100]      <1>       mov    eax, [createfile_FFCluster]
7486 0000C5D5 E846040000        <1>       call   load_FAT_sub_directory
7487 0000C5DA 72D3              <1>       jc     short loc_createfile_anc_retn
7488                            <1>
7489                            <1> pass_createfile_add_new_subdir_cluster:
7490                            <1>       ;movzx eax, word [esi+LD_BPB+BytesPerSec]
7491 0000C5DC 0FB705[D48C0100]  <1>       movzx  eax, word [createfile_BytesPerSec] ; 23/03/2016
7492 0000C5E3 F7E1              <1>       mul    ecx ; ecx = directory buffer sector count
7493 0000C5E5 89C1              <1>       mov    ecx, eax
7494 0000C5E7 C1E902            <1>       shr    ecx, 2 ; dword count
7495 0000C5EA 29C0              <1>       sub    eax, eax ; 0
7496 0000C5EC F3AB              <1>       rep    stosd
7497                            <1>       ;
7498 0000C5EE C605[94880100]02  <1>       mov    byte [DirBuff_ValidData], 2
7499 0000C5F5 E890EFFFFF        <1>       call   save_directory_buffer
7500 0000C5FA 72B3              <1>       jc     short loc_createfile_anc_retn
7501                            <1>
7502                            <1> loc_createfile_save_added_subdir_cluster:
7503 0000C5FC A1[C08C0100]      <1>       mov    eax, [createfile_LastDirCluster]
7504 0000C601 8B0D[BC8C0100]    <1>       mov    ecx, [createfile_FFCluster]
7505 0000C607 E858050000        <1>       call   update_cluster
7506 0000C60C 7304              <1>       jnc    short loc_createfile_save_fat_buffer_0
7507 0000C60E 09C0              <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
7508 0000C610 751A              <1>       jnz    short loc_createfile_save_fat_buffer_stc_retn
7509                            <1>
7510                            <1> loc_createfile_save_fat_buffer_0:
7511 0000C612 A1[BC8C0100]      <1>       mov    eax, [createfile_FFCluster]
7512 0000C617 A3[C08C0100]      <1>       mov    [createfile_LastDirCluster], eax
7513 0000C61C B9FFFFFF0F        <1>       mov    ecx, 0FFFFFFFh ; 28 bit
7514 0000C621 E83E050000        <1>       call   update_cluster
7515 0000C626 7306              <1>       jnc    short loc_createfile_save_fat_buffer_1
7516 0000C628 09C0              <1>       or     eax, eax ; Was it free cluster
7517 0000C62A 7402              <1>       jz     short loc_createfile_save_fat_buffer_1
7518                            <1>
7519                            <1> loc_createfile_save_fat_buffer_stc_retn:
7520 0000C62C F9                <1>       stc
7521                            <1> loc_createfile_save_fat_buffer_retn:
7522                            <1> loc_createfile_gffc_2_stc_retn:
7523 0000C62D C3                <1>       retn
7524                            <1>
7525                            <1> loc_createfile_save_fat_buffer_1:
7526                            <1>       ; byte [FAT_BuffValidData] = 2
7527 0000C62E E8EE070000        <1>       call   save_fat_buffer
7528 0000C633 72F8              <1>       jc     short loc_createfile_save_fat_buffer_retn
7529                            <1>
7530 0000C635 803D[8A880100]01  <1>       cmp    byte [FAT_ClusterCounter], 1
7531 0000C63C 7222              <1>       jb     short loc_createfile_save_fat_buffer_2
7532                            <1>
7533                            <1>       ; ESI = Logical DOS Drive Description Table address
7534 0000C63E A1[8A880100]      <1>       mov    eax, [FAT_ClusterCounter]
7535                            <1>
7536 0000C643 C605[8A880100]00  <1>       mov    byte [FAT_ClusterCounter], 0 ; 21/03/2016
7537                            <1>
7538 0000C64A 66BB01FF          <1>       mov    bx, 0FF01h ; add free clusters
7539 0000C64E E863080000        <1>       call   calculate_fat_freespace
7540                            <1>
7541                            <1>       ;inc   eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
7542                            <1>       ;jnz   short loc_createfile_save_fat_buffer_2
```

```
7543                                <1>
7544                                <1>       ; ecx > 0 -> Recalculation is needed
7545 0000C653 09C9                 <1>       or     ecx, ecx
7546 0000C655 7409                 <1>       jz     short loc_createfile_save_fat_buffer_2
7547                                <1>
7548 0000C657 66BB00FF             <1>       mov    bx, 0FF00h ; ; recalculate free space
7549 0000C65B E856080000           <1>       call   calculate_fat_freespace
7550                                <1>
7551                                <1> loc_createfile_save_fat_buffer_2:
7552                                <1>       ;call  update_parent_dir_lmdt
7553                                <1>
7554                                <1> loc_createfile_gffc_2:
7555 0000C660 E82C040000           <1>       call   get_first_free_cluster
7556 0000C665 72C6                 <1>       jc     short loc_createfile_gffc_2_stc_retn
7557                                <1>
7558 0000C667 A3[BC8C0100]         <1>       mov    [createfile_FFCluster], eax
7559                                <1>
7560 0000C66C A1[C08C0100]         <1>       mov    eax, [createfile_LastDirCluster]
7561                                <1>
7562 0000C671 E8AA030000           <1>       call   load_FAT_sub_directory
7563 0000C676 72B5                 <1>       jc     short loc_createfile_gffc_2_stc_retn
7564                                <1>
7565 0000C678 BF00000800           <1>       mov    edi, Directory_Buffer
7566                                <1>
7567 0000C67D 6629DB               <1>       sub    bx, bx ; directory entry index/number = 0
7568                                <1>
7569 0000C680 56                   <1>       push   esi ; * ; 23/03/2016
7570                                <1>
7571                                <1> loc_createfile_set_ff_dir_entry:
7572 0000C681 66891D[CE8C0100]     <1>       mov    [createfile_DirIndex], bx
7573                                <1>
7574                                <1>         ; EDI = Directory entry address
7575 0000C688 8B35[B08C0100]       <1>       mov    esi, [createfile Name Offset]
7576 0000C68E A1[BC8C0100]         <1>       mov    eax, [createfile_FFCluster]
7577 0000C693 A3[C48C0100]         <1>       mov    [createfile_Cluster], eax ; 24/03/2016
7578 0000C698 B5FF                 <1>       mov    ch, 0FFh
7579 0000C69A 8A0D[CC8C0100]       <1>         mov  cl, [createfile_attrib] ; file attributes
7580                                <1>       ; CH > 0 -> File size is in [EBX]
7581 0000C6A0 BB[B88C0100]         <1>       mov    ebx, createfile_size
7582                                <1>
7583 0000C6A5 E803EEFFFF           <1>       call   make_directory_entry
7584                                <1>
7585 0000C6AA 5E                   <1>       pop    esi ; * ; ESI = Logical Dos Drv Desc. Table address
7586                                <1>
7587 0000C6AB C605[94880100]02     <1>       mov    byte [DirBuff_ValidData], 2
7588 0000C6B2 E8D3EEFFFF           <1>       call   save_directory_buffer
7589 0000C6B7 7221                 <1>       jc     short loc_createfile_set_ff_dir_entry_retn
7590                                <1>
7591 0000C6B9 C605[D78C0100]01     <1>       mov    byte [createfile_UpdatePDir], 1 ; 31/03/2016
7592                                <1>
7593                                <1> loc_createfile_get_set_write_file_cluster:
7594 0000C6C0 A1[B88C0100]         <1>       mov    eax, [createfile_size]
7595 0000C6C5 09C0                 <1>       or     eax, eax
7596 0000C6C7 7570                 <1>       jnz    short loc_createfile_get_set_wfc_cont
7597 0000C6C9 40                   <1>       inc    eax
7598                                <1>       ; 23/03/2016
7599 0000C6CA 0FB61D[CD8C0100]     <1>       movzx  ebx, byte [createfile_SecPerClust]
7600                                <1>       ;movzx ecx, word [esi+LD_BPB+BytesPerSec] ; 512
7601 0000C6D1 0FB70D[D48C0100]     <1>         movzx   ecx, word [createfile_BytesPerSec] ; 512
7602 0000C6D8 EB7C                 <1>       jmp    loc_createfile_set_cluster_count
7603                                <1>
7604                                <1> loc_createfile_set_ff_dir_entry_retn:
7605 0000C6DA C3                   <1>       retn
7606                                <1>
7607                                <1> loc_createfile_write_fcluster_to_disk:
7608 0000C6DB 034668               <1>       add    eax, [esi+LD_DATABegin] ; convert to physical address
7609 0000C6DE BB00000700           <1>       mov    ebx, Cluster_Buffer
7610                                <1>       ; ESI = Logical DOS Drv. Desc. Tbl. address
7611                                <1>       ; EAX = Disk address
7612                                <1>       ; EBX = Sector Buffer
7613                                <1>       ; ECX = sectors per cluster
7614 0000C6E3 E8A45E0000           <1>       call   disk_write
7615 0000C6E8 7211                 <1>       jc     short loc_createfile_dsk_wr_err
7616                                <1>
7617                                <1> loc_createfile_update_fat_cluster:
7618                                <1>       ; 21/03/2016
7619 0000C6EA 803D[D68C0100]00     <1>       cmp    byte [createfile_wfc], 0
7620 0000C6F1 7712                 <1>       ja     short loc_createfile_update_fat_cluster_n1
7621                                <1>
7622 0000C6F3 FE05[D68C0100]       <1>       inc    byte [createfile_wfc] ; 1
7623 0000C6F9 EB24                 <1>       jmp    short loc_createfile_update_fat_cluster_n2
7624                                <1>
7625                                <1> loc_createfile_dsk_wr_err:
7626                                <1>       ; 16/10/2016 (1Dh -> 18)
7627                                <1>       ; 23/03/2016
7628 0000C6FB B812000000           <1>       mov    eax, 18 ; Drive not ready or write error !
7629 0000C700 E9BD000000           <1>       jmp    loc_createfile_stc_retn
7630                                <1>
7631                                <1> loc_createfile_update_fat_cluster_n1:
7632 0000C705 A1[C88C0100]         <1>       mov    eax, [createfile_PCluster]
7633 0000C70A 8B0D[C48C0100]       <1>       mov    ecx, [createfile_Cluster]
7634 0000C710 E84F040000           <1>       call   update_cluster
7635 0000C715 7308                 <1>       jnc    short loc_createfile_update_fat_cluster_n2
7636 0000C717 09C0                 <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
7637 0000C719 0F85A3000000         <1>       jnz    loc_createfile_stc_retn
7638                                <1>
7639                                <1> loc_createfile_update_fat_cluster_n2:
7640 0000C71F A1[C48C0100]         <1>       mov    eax, [createfile_Cluster]
7641 0000C724 B9FFFFFF0F           <1>       mov    ecx, 0FFFFFFFh
7642 0000C729 E836040000           <1>       call   update_cluster
7643 0000C72E 734E                 <1>       jnc    short loc_createfile_save_fat_buffer_3
7644 0000C730 09C0                 <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
7645 0000C732 744A                 <1>       jz     short loc_createfile_save_fat_buffer_3
7646                                <1>
7647                                <1> loc_createfile_upd_fat_fcluster_stc_retn:
```

```
7648 0000C734 E989000000          <1>       jmp    loc_createfile_stc_retn
7649                              <1>
7650                              <1> loc_createfile_get_set_wfc_cont:
7651                              <1>       ;movzx ecx, word [esi+LD_BPB+BytesPerSec] ; 512
7652 0000C739 0FB70D[D48C0100]    <1>       movzx  ecx, word [createfile_BytesPerSec] ; 512
7653 0000C740 01C8                <1>       add    eax, ecx
7654 0000C742 48                  <1>       dec    eax  ; add eax, 511
7655 0000C743 29D2                <1>       sub    edx, edx
7656 0000C745 F7F1                <1>       div    ecx
7657 0000C747 0FB61D[CD8C0100]    <1>       movzx  ebx, byte [createfile_SecPerClust]
7658 0000C74E 01D8                <1>       add    eax, ebx
7659 0000C750 48                  <1>       dec    eax  ; add eax, SecPerClust - 1
7660 0000C751 6631D2              <1>       xor    dx, dx
7661 0000C754 F7F3                <1>       div    ebx
7662                              <1>
7663                              <1> loc_createfile_set_cluster_count:
7664 0000C756 A3[D08C0100]        <1>       mov    [createfile_CCount], eax
7665                              <1>
7666 0000C75B BF00000700          <1>       mov    edi, Cluster_Buffer
7667 0000C760 89C8                <1>       mov    eax, ecx ; Bytes per Sector
7668 0000C762 F7E3                <1>       mul    ebx ; Sectors per Cluster
7669                              <1>       ; EAX = Bytes per Cluster
7670 0000C764 89C1                <1>       mov    ecx, eax
7671 0000C766 C1E902              <1>       shr    ecx, 2 ; dword count
7672 0000C769 31C0                <1>       xor    eax, eax
7673 0000C76B F3AB                <1>       rep    stosd ; clear cluster buffer
7674                              <1>
7675 0000C76D A1[C48C0100]        <1>       mov    eax, [createfile_Cluster] ; 24/03/2016
7676                              <1>
7677 0000C772 89D9                <1>       mov    ecx, ebx
7678                              <1>
7679                              <1> loc_createfile_get_set_wf_fclust_cont:
7680 0000C774 83E802              <1>       sub    eax, 2
7681 0000C777 F7E1                <1>       mul    ecx
7682                              <1>       ; EAX = Logical DOS disk address (offset)
7683 0000C779 E95DFFFFFF          <1>       jmp    loc_createfile_write_fcluster_to_disk
7684                              <1>
7685                              <1> loc_createfile_save_fat_buffer_3:
7686                              <1>       ; byte [FAT_BuffValidData] = 2
7687 0000C77E E89E060000          <1>       call   save_fat_buffer
7688 0000C783 723D                <1>       jc     loc_createfile_stc_retn
7689                              <1>
7690                              <1>       ; 21/03/2016
7691 0000C785 803D[8A880100]01    <1>       cmp    byte [FAT_ClusterCounter], 1
7692 0000C78C 721B                <1>       jb     short loc_createfile_save_fat_buffer_4
7693                              <1>
7694                              <1>       ; ESI = Logical DOS Drive Description Table address
7695 0000C78E A1[8A880100]        <1>       mov    eax, [FAT_ClusterCounter]
7696 0000C793 66BB01FF            <1>       mov    bx, 0FF01h ; add free clusters
7697 0000C797 E81A070000          <1>       call   calculate_fat_freespace
7698                              <1>
7699                              <1>       ;inc   eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
7700                              <1>       ;jnz   short loc_createfile_save_fat_buffer_4
7701                              <1>
7702                              <1>       ; ecx > 0 -> Recalculation is needed
7703 0000C79C 09C9                <1>       or     ecx, ecx
7704 0000C79E 7409                <1>       jz     short loc_createfile_save_fat_buffer_4
7705                              <1>
7706 0000C7A0 66BB00FF            <1>       mov    bx, 0FF00h ; ; recalculate free space
7707 0000C7A4 E80D070000          <1>       call   calculate_fat_freespace
7708                              <1>
7709                              <1> loc_createfile_save_fat_buffer_4:
7710 0000C7A9 FF0D[D08C0100]      <1>       dec    dword [createfile_CCount]
7711                              <1>       ;jz    short loc_createfile_upd_dir_modif_date_time
7712 0000C7AF 743F                <1>       jz     short loc_createfile_stc_retn_cc ; 31/03/2016
7713                              <1>
7714                              <1> loc_createfile_get_set_write_next_cluster:
7715 0000C7B1 E8DB020000          <1>       call   get_first_free_cluster
7716 0000C7B6 720A                <1>       jc     short loc_createfile_stc_retn
7717                              <1>
7718                              <1> loc_createfile_get_set_write_next_cluster_1:
7719 0000C7B8 83F8FF              <1>       cmp    eax, 0FFFFFFFFh
7720 0000C7BB 7213                <1>       jb     short loc_createfile_get_set_write_next_cluster_2
7721                              <1>
7722                              <1> loc_createfile_wnc_insufficient_disk_space:
7723 0000C7BD B827000000          <1>       mov    eax, 27h ; Insufficient disk space
7724                              <1>
7725                              <1> loc_createfile_stc_retn:
7726 0000C7C2 803D[D68C0100]01    <1>       cmp    byte [createfile_wfc], 1
7727 0000C7C9 7324                <1>       jnb    short loc_createfile_err_retn
7728 0000C7CB C3                  <1>       retn
7729                              <1>
7730                              <1> loc_createfile_wnc_inv_format_retn:
7731                              <1>       ;mov   eax, 28
7732 0000C7CC B01C                <1>       mov    al, 28 ; Invalid format
7733 0000C7CE EBF2                <1>       jmp    short loc_createfile_stc_retn
7734                              <1>
7735                              <1> loc_createfile_get_set_write_next_cluster_2:
7736 0000C7D0 83F802              <1>       cmp    eax, 2
7737 0000C7D3 72F7                <1>       jb     short loc_createfile_wnc_inv_format_retn
7738                              <1>
7739                              <1> loc_createfile_get_set_write_next_cluster_3:
7740 0000C7D5 8B0D[C48C0100]      <1>       mov    ecx, [createfile_Cluster]
7741 0000C7DB A3[C48C0100]        <1>       mov    [createfile_Cluster], eax
7742 0000C7E0 890D[C88C0100]      <1>       mov    [createfile_PCluster], ecx
7743 0000C7E6 0FB60D[CD8C0100]    <1>       movzx  ecx, byte [createfile_SecPerClust]
7744 0000C7ED EB85                <1>       jmp    short loc_createfile_get_set_wf_fclust_cont
7745                              <1>
7746                              <1> loc_createfile_err_retn:
7747 0000C7EF F9                  <1>       stc
7748                              <1>
7749                              <1> ;loc_createfile_upd_dir_modif_date_time:
7750                              <1> loc_createfile_stc_retn_cc: ; 31/03/2016
7751 0000C7F0 9C                  <1>       pushf  ; cpu is here for an error return or completion
7752 0000C7F1 50                  <1>       push   eax ; error code if cf = 1
```

```
7753                                    <1>
7754                                    <1>     ;call  update_parent_dir_lmdt
7755                                    <1>
7756                                    <1> ;loc_createfile_stc_retn_cc:
7757 0000C7F2 A1[8A880100]             <1>     mov    eax, [FAT_ClusterCounter]
7758 0000C7F7 09C0                     <1>     or     eax, eax
7759 0000C7F9 741A                     <1>     jz     short loc_createfile_stc_retn_pop_eax
7760 0000C7FB 8A3D[6E810100]           <1>     mov    bh, [Current_Drv]
7761 0000C801 B301                     <1>     mov    bl, 01h ; BL = 1 -> add clusters
7762                                    <1>     ; NOTE: EAX value will be added to Free Cluster Count
7763                                    <1>     ; (If EAX value is negative, Free Cluster Count will be decreased)
7764 0000C803 E8AE060000               <1>     call   calculate_fat_freespace
7765                                    <1>       ; ESI = Logical DOS Drive Description Table Address
7766                                    <1>       ;jc  short loc_createfile_stc_retn_pop_eax_cf
7767 0000C808 21C9                     <1>     and    ecx, ecx ; cx = 0 -> valid free sector count
7768 0000C80A 7409                     <1>     jz     short loc_createfile_stc_retn_pop_eax
7769                                    <1>
7770                                    <1> loc_createfile_stc_retn_recalc_FAT_freespace:
7771 0000C80C 66BB00FF                 <1>     mov    bx, 0FF00h ; bh = 0FFh ->
7772                                    <1>     ; ESI = Logical DOS Drv DT Addr
7773                                    <1>     ; BL = 0 -> Recalculate
7774 0000C810 E8A1060000               <1>     call   calculate_fat_freespace
7775                                    <1>
7776                                    <1> loc_createfile_stc_retn_pop_eax:
7777 0000C815 58                       <1>     pop    eax
7778 0000C816 9D                       <1>     popf
7779 0000C817 7218                     <1>     jc     short loc_createfile_retn
7780                                    <1>
7781                                    <1> loc_createfile_retn_fcluster:
7782 0000C819 A1[BC8C0100]             <1>     mov    eax, [createfile_FFCluster]
7783 0000C81E BB[B88C0100]             <1>     mov    ebx, createfile_size
7784                                    <1>     ;movzx ecx, byte [esi+LD_BPB+SecPerClust]
7785 0000C823 0FB60D[CD8C0100]         <1>     movzx  ecx, byte [createfile_SecPerClust] ; 23/03/2016
7786 0000C82A 0FB715[CE8C0100]         <1>     movzx  edx, word [createfile_DirIndex]
7787                                    <1>
7788                                    <1> loc_createfile_retn:
7789 0000C831 C3                       <1>     retn
7790                                    <1>
7791                                    <1> create_fs_file:
7792                                    <1>     ; temporary (21/03/2016)
7793 0000C832 C3                       <1>     retn
7794                                    <1>
7795                                    <1> delete_fs_file:
7796                                    <1>     ; temporary (28/02/2016)
7797 0000C833 C3                       <1>     retn
7798                                    <1>
7799                                    <1> rename_fs_file_or_directory:
7800 0000C834 C3                       <1>     retn
7801                                    <1>
7802                                    <1> make_fs_directory:
7803                                    <1>     ; temporary (21/02/2016)
7804 0000C835 C3                       <1>     retn
7805                                    <1>
7806                                    <1> add_new_fs_section:
7807                                    <1>     ; temporary (11/03/2016)
7808 0000C836 C3                       <1>     retn
7809                                    <1>
7810                                    <1> delete_fs_directory_entry:
7811                                    <1>     ; temporary (11/03/2016)
7812 0000C837 C3                       <1>     retn
7813                                    <1>
7814                                    <1> csftdf2_read_fs_file_sectors:
7815                                    <1>     ; temporary (19/03/2016)
7816 0000C838 C3                       <1>     retn
7817                                    <1>
7818                                    <1> csftdf2_write_fs_file_sectors:
7819                                    <1>     ; temporary (19/03/2016)
7820 0000C839 C3                       <1>     retn
3110                                        %include 'trdosk5.s' ; 24/01/2016
3111                                    <1> ; ****************************************************************************
3112                                    <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - File System Procedures : trdosk5s
3113                                    <1> ; ----------------------------------------------------------------------------
3114                                    <1> ; Last Update: 23/10/2016
3115                                    <1> ; ----------------------------------------------------------------------------
3116                                    <1> ; Beginning: 24/01/2016
3117                                    <1> ; ----------------------------------------------------------------------------
3118                                    <1> ; Assembler: NASM version 2.11 (trdos386.s)
3119                                    <1> ; ----------------------------------------------------------------------------
3120                                    <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
3121                                    <1> ; DRV_FAT.ASM (21/08/2011)
3122                                    <1> ; ****************************************************************************
3123                                    <1> ; DRV_FAT.ASM (c) 2005-2011 Erdogan TAN [ 07/07/2009 ] Last Update: 21/08/2011
3124                                    <1>
3125                                    <1> get_next_cluster:
3126                                    <1>       ; 15/10/2016
3127                                    <1>       ; 23/03/2016
3128                                    <1>       ; 01/02/2016 (TRDOS 386 =  TRDOS v2.0)
3129                                    <1>       ; 05/07/2011
3130                                    <1>       ; 07/07/2009
3131                                    <1>       ; 2005
3132                                    <1>       ; INPUT ->
3133                                    <1>       ;     EAX = Cluster Number (32 bit)
3134                                    <1>       ;     ESI = Logical DOS Drive Parameters Table
3135                                    <1>       ; OUTPUT ->
3136                                    <1>       ;     cf = 0 -> No Error, EAX valid
3137                                    <1>       ;     cf = 1 & EAX = 0 -> End Of Cluster Chain
3138                                    <1>       ;     cf = 1 & EAX > 0 -> Error
3139                                    <1>       ;     ECX = Current/Previous cluster (if CF = 0)
3140                                    <1>       ;     EAX = Next Cluster Number (32 bit)
3141                                    <1>       ;
3142                                    <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
3143                                    <1>
3144 0000C83A A3[7E880100]             <1>       mov    [FAT_CurrentCluster], eax
3145                                    <1> check_next_cluster_fat_type:
3146 0000C83F 29D2                     <1>       sub    edx, edx ; 0
```

```
3147 0000C841 807E0302          <1>       cmp    byte [esi+LD_FATType], 2
3148 0000C845 7250              <1>       jb     short get_FAT12_next_cluster
3149 0000C847 0F87AF000000      <1>       ja     get_FAT32_next_cluster
3150                            <1> get_FAT16_next_cluster:
3151 0000C84D BB00030000        <1>       mov    ebx, 300h ;768
3152 0000C852 F7F3              <1>       div    ebx
3153                            <1>       ; EAX = Count of 3 FAT sectors
3154                            <1>       ; EDX = Cluster Offset (< 768)
3155 0000C854 66D1E2            <1>       shl    dx, 1 ; Multiply by 2
3156 0000C857 89D3              <1>       mov    ebx, edx ; Byte Offset
3157 0000C859 81C3001C0900      <1>       add    ebx, FAT_Buffer
3158 0000C85F 66BA0300          <1>       mov    dx, 3
3159 0000C863 F7E2              <1>       mul    edx
3160                            <1>       ; EAX = FAT Sector (<= 256)
3161                            <1>       ; EDX = 0
3162 0000C865 8A0E              <1>       mov    cl, [esi+LD_Name]
3163 0000C867 803D[82880100]00  <1>       cmp    byte [FAT_BuffValidData], 0
3164 0000C86E 0F86CC000000      <1>       jna    load_FAT_sectors0
3165 0000C874 3A0D[83880100]    <1>       cmp    cl, [FAT_BuffDrvName]
3166 0000C87A 0F85C0000000      <1>       jne    load_FAT_sectors0
3167 0000C880 3B05[86880100]    <1>       cmp    eax, [FAT_BuffSector]
3168 0000C886 0F85BA000000      <1>       jne    load_FAT_sectors1
3169                            <1>       ;movzx eax, word [ebx]
3170 0000C88C 668B03            <1>       mov    ax, [ebx]
3171                            <1>       ; 01/02/2016
3172                            <1>       ; DRV_FAT.ASM (21/08/2011) had a FATal bug here !
3173                            <1>       ; (cmp ah, 0Fh) ! (ax >= FF7h)
3174                            <1>       ; (how can i do a such mistake!?)
3175                            <1>       ;cmp   al, 0F7h
3176                            <1>       ;jb    short loc_pass_gnc_FAT16_eoc_check
3177                            <1>       ;cmp   ah, 0FFh
3178                            <1>       ;jb    short loc_pass_gnc_FAT16_eoc_check
3179 0000C88F 6683F8F7          <1>       cmp    ax, 0FFF7h
3180 0000C893 725A              <1>       jb     short loc_pass_gnc_FAT16_eoc_check
3181                            <1>       ; ax >= FFF7h (cluster 0002h to FFF6h is valid, in use)
3182 0000C895 EB56              <1>       jmp    short loc_pass_gnc_FAT16_eoc_check_xor_eax
3183                            <1>
3184                            <1> get_FAT12_next_cluster:
3185 0000C897 BB00040000        <1>       mov    ebx, 400h ;1024
3186 0000C89C F7F3              <1>       div    ebx
3187                            <1>       ; EAX = Count of 3 FAT sectors
3188                            <1>       ; EDX = Cluster Offset (< 1024)
3189 0000C89E 6650              <1>       push   ax
3190 0000C8A0 66B80300          <1>       mov    ax, 3
3191 0000C8A4 66F7E2            <1>       mul    dx     ; Multiply by 3
3192 0000C8A7 66D1E8            <1>       shr    ax, 1  ; Divide by 2
3193 0000C8AA 6689C3            <1>       mov    bx, ax       ; Byte Offset
3194 0000C8AD 81C3001C0900      <1>       add    ebx, FAT_Buffer
3195 0000C8B3 6658              <1>       pop    ax
3196 0000C8B5 66BA0300          <1>       mov    dx, 3
3197 0000C8B9 F7E2              <1>       mul    edx
3198                            <1>       ; EAX = FAT Sector (<= 12)
3199                            <1>       ; EDX = 0
3200 0000C8BB 8A0E              <1>       mov    cl, [esi+LD_Name]
3201 0000C8BD 803D[82880100]00  <1>       cmp    byte [FAT_BuffValidData], 0
3202 0000C8C4 767A              <1>       jna    short load_FAT_sectors0
3203 0000C8C6 3A0D[83880100]    <1>       cmp    cl, [FAT_BuffDrvName]
3204 0000C8CC 7572              <1>       jne    short load_FAT_sectors0
3205 0000C8CE 3B05[86880100]    <1>       cmp    eax, [FAT_BuffSector]
3206 0000C8D4 7570              <1>       jne    short load_FAT_sectors1
3207 0000C8D6 A1[7E880100]      <1>       mov    eax, [FAT_CurrentCluster]
3208 0000C8DB 66D1E8            <1>       shr    ax, 1
3209                            <1>       ;movzx eax, word [ebx]
3210 0000C8DE 668B03            <1>       mov    ax, [ebx]
3211 0000C8E1 7314              <1>       jnc    short get_FAT12_nc_even
3212 0000C8E3 66C1E804          <1>       shr    ax, 4
3213                            <1> loc_gnc_fat12_eoc_check:
3214                            <1>       ;cmp   al, 0F7h
3215                            <1>       ;jb    short loc_pass_gnc_FAT16_eoc_check
3216                            <1>       ;cmp   ah, 0Fh
3217                            <1>       ;jb    short loc_pass_gnc_FAT16_eoc_check
3218 0000C8E7 663DF70F          <1>       cmp    ax, 0FF7h
3219 0000C8EB 7202              <1>       jb     short loc_pass_gnc_FAT16_eoc_check
3220                            <1>       ; ax >= FF7h (cluster 0002h to FF6h is valid, in use)
3221                            <1>
3222                            <1> loc_pass_gnc_FAT16_eoc_check_xor_eax:
3223 0000C8ED 31C0              <1>       xor    eax, eax ; 0
3224                            <1> loc_pass_gnc_FAT16_eoc_check:
3225                            <1> loc_pass_gnc_FAT32_eoc_check:
3226 0000C8EF 8B0D[7E880100]    <1>       mov    ecx, [FAT_CurrentCluster]
3227 0000C8F5 F5                <1>       cmc
3228 0000C8F6 C3                <1>       retn
3229                            <1>
3230                            <1> get_FAT12_nc_even:
3231 0000C8F7 80E40F            <1>       and    ah, 0Fh
3232 0000C8FA EBEB              <1>       jmp    short loc_gnc_fat12_eoc_check
3233                            <1>
3234                            <1> get_FAT32_next_cluster:
3235 0000C8FC BB80010000        <1>       mov    ebx, 180h ;384
3236 0000C901 F7F3              <1>       div    ebx
3237                            <1>       ; EAX = Count of 3 FAT sectors
3238                            <1>       ; EDX = Cluster Offset (< 384)
3239 0000C903 66C1E202          <1>       shl    dx, 2  ; Multiply by 4
3240 0000C907 89D3              <1>       mov    ebx, edx ; Byte Offset
3241 0000C909 81C3001C0900      <1>       add    ebx, FAT_Buffer
3242 0000C90F 66BA0300          <1>       mov    dx, 3
3243 0000C913 F7E2              <1>       mul    edx
3244                            <1>         ; EAX = FAT Sector (<= 2097152) ; (FFFFFF7h * 4) / 512
3245                            <1>       ;      for 32KB cluster size:
3246                            <1>       ;      EAX <= 1024 = (4GB / 32KB) * 4) / 512
3247                            <1>       ; EDX = 0
3248 0000C915 8A0E              <1>       mov    cl, [esi+LD_Name]
3249 0000C917 803D[82880100]00  <1>       cmp    byte [FAT_BuffValidData], 0
3250 0000C91E 7620              <1>       jna    short load_FAT_sectors0
3251 0000C920 3A0D[83880100]    <1>       cmp    cl, [FAT_BuffDrvName]
```

```
3252 0000C926 7518               <1>       jne    short load_FAT_sectors0
3253 0000C928 3B05[86880100]     <1>       cmp    eax, [FAT_BuffSector] ; 0, 3, 6, 9 ...
3254 0000C92E 7516               <1>       jne    short load_FAT_sectors1
3255 0000C930 8B03               <1>       mov    eax, [ebx]
3256 0000C932 25FFFFFF0F         <1>       and    eax, 0FFFFFFFh ; 28 bit Cluster
3257 0000C937 3DF7FFFF0F         <1>       cmp    eax, 0FFFFFF7h
3258 0000C93C 72B1               <1>       jb     short loc_pass_gnc_FAT32_eoc_check
3259                             <1>       ; eax >= FFFFFF7h (cluster 0002h to FFFFFF6h is valid)
3260 0000C93E EBAD               <1>       jmp    short loc_pass_gnc_FAT16_eoc_check_xor_eax
3261                             <1>
3262                             <1> load_FAT_sectors0:
3263 0000C940 880D[83880100]     <1>       mov    [FAT_BuffDrvName], cl
3264                             <1> load_FAT_sectors1:
3265 0000C946 A3[86880100]       <1>       mov    [FAT_BuffSector], eax
3266 0000C94B 89C3               <1>       mov    ebx, eax
3267 0000C94D 034660             <1>       add    eax, [esi+LD_FATBegin]
3268 0000C950 807E0302           <1>       cmp    byte [esi+LD_FATType], 2
3269 0000C954 7706               <1>       ja     short load_FAT_sectors3
3270 0000C956 0FB74E1C           <1>       movzx  ecx, word [esi+LD_BPB+BPB_FATSz16]
3271 0000C95A EB03               <1>       jmp    short load_FAT_sectors4
3272                             <1> load_FAT_sectors3:
3273 0000C95C 8B4E2A             <1>       mov    ecx, [esi+LD_BPB+BPB_FATSz32]
3274                             <1> load_FAT_sectors4:
3275 0000C95F 29D9               <1>       sub    ecx, ebx ; [FAT_BuffSector]
3276 0000C961 83F903             <1>       cmp    ecx, 3
3277 0000C964 7605               <1>       jna    short load_FAT_sectors5
3278 0000C966 B903000000         <1>       mov    ecx, 3
3279                             <1> load_FAT_sectors5:
3280 0000C96B BB001C0900         <1>       mov    ebx, FAT_Buffer
3281 0000C970 E8265C0000         <1>       call   disk_read
3282 0000C975 730D               <1>       jnc    short load_FAT_sectors_ok
3283                             <1>       ; 15/10/2016 (15h -> 17)
3284                             <1>       ; 23/03/2016 (15h)
3285 0000C977 B811000000         <1>       mov    eax, 17 ; Drive not ready or read error
3286 0000C97C C605[82880100]00   <1>       mov    byte [FAT_BuffValidData], 0
3287 0000C983 C3                 <1>       retn
3288                             <1> load_FAT_sectors_ok:
3289 0000C984 C605[82880100]01   <1>       mov    byte [FAT_BuffValidData], 1
3290 0000C98B A1[7E880100]       <1>       mov    eax, [FAT_CurrentCluster]
3291 0000C990 E9AAFEFFFF         <1>       jmp    check_next_cluster_fat_type
3292                             <1>
3293                             <1> load_FAT_root_directory:
3294                             <1>       ; 23/10/2016
3295                             <1>       ; 15/10/2016
3296                             <1>       ; 07/02/2016
3297                             <1>       ; 02/02/2016
3298                             <1>       ; 01/02/2016 (TRDOS 386 =  TRDOS v2.0)
3299                             <1>       ; 21/05/2011
3300                             <1>       ; 22/08/2009
3301                             <1>       ;
3302                             <1>       ; INPUT ->
3303                             <1>       ;      ESI = Logical DOS Drive Description Table
3304                             <1>       ; OUTPUT ->
3305                             <1>       ;      cf = 1 -> Root directory could not be loaded
3306                             <1>       ;          EAX > 0 -> Error number
3307                             <1>       ;      cf = 0 -> EAX = 0
3308                             <1>       ;      ECX = Directory buffer size in sectors (CL)
3309                             <1>       ;      EBX = Directory buffer address
3310                             <1>       ;      NOTE: DirBuffer_Size is in bytes ! (word)
3311                             <1>       ;
3312                             <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
3313                             <1>
3314                             <1>       ; NOTE: Only for FAT12 and FAT16 file systems !
3315                             <1>       ; (FAT32 fs root dir must be loaded as sub directory)
3316                             <1>
3317 0000C995 8A1E               <1>       mov    bl, [esi+LD_Name]
3318 0000C997 8A7E03             <1>       mov    bh, [esi+LD_FATType]
3319                             <1>
3320                             <1>       ;mov   [DirBuff_DRV], bl
3321                             <1>       ;mov   [DirBuff_FATType], bh
3322 0000C99A 66891D[92880100]   <1>       mov    [DirBuff_DRV], bx
3323                             <1>
3324                             <1>       ;cmp   bh, 2
3325                             <1>       ;ja    short load_FAT32_root_dir0 ; FAT32 root dir
3326                             <1>
3327                             <1> load_FAT_root_dir0: ; 23/10/2016
3328 0000C9A1 0FB75617           <1>       movzx  edx, word [esi+LD_BPB+RootDirEnts]
3329                             <1>
3330                             <1>       ;or    dx, dx ; 0 for FAT32 file systems
3331                             <1>       ;jz    short load_FAT32_root_dir0 ; FAT32 root dir
3332                             <1>
3333 0000C9A5 6681FA0002           <1>       cmp    dx, 512 ; Number of Root Dir Entries
3334 0000C9AA 7414               <1>       je     short lrd_mov_ecx_32
3335 0000C9AC 89D0               <1>       mov    eax, edx
3336                             <1>       ; 23/10/2016
3337 0000C9AE 89C1               <1>       mov    ecx, eax
3338 0000C9B0 6683C10F           <1>       add    cx, 15 ; round up
3339 0000C9B4 66C1E904           <1>       shr    cx, 4  ; 16 entries per sector (512/32)
3340                             <1>       ; ecx = Root directory size in sectors
3341 0000C9B8 66C1E005           <1>       shl    ax, 5 ; Root directory size in bytes
3342 0000C9BC 664A               <1>       dec    dx    ; Last entry number of root dir
3343                             <1>       ; cx = Dir Buffer sector count
3344 0000C9BE EB0B               <1>       jmp    short lrd_check_dir_buffer
3345                             <1>
3346                             <1> lrd_mov_ecx_32:
3347 0000C9C0 B920000000         <1>       mov    ecx, 32
3348 0000C9C5 664A               <1>       dec    dx ; 511
3349 0000C9C7 66B80040           <1>       mov    ax, 32*512
3350                             <1>
3351                             <1> lrd_check_dir_buffer:
3352 0000C9CB 29DB               <1>       sub    ebx, ebx ; 0
3353 0000C9CD 881D[94880100]     <1>       mov    [DirBuff_ValidData], bl ; 0
3354 0000C9D3 668915[97880100]   <1>       mov    [DirBuff_LastEntry], dx
3355 0000C9DA 891D[99880100]     <1>       mov    [DirBuff_Cluster], ebx ; 0
3356 0000C9E0 66A3[9D880100]     <1>       mov    [DirBuffer_Size], ax
```

```
3357                                <1>
3358 0000C9E6 8B4664                <1>       mov    eax, [esi+LD_ROOTBegin]
3359                                <1> read_directory:
3360 0000C9E9 BB00000800            <1>       mov    ebx, Directory_Buffer
3361 0000C9EE 51                    <1>       push   ecx ; Directory buffer sector count
3362 0000C9EF 53                    <1>       push   ebx
3363 0000C9F0 E8A65B0000            <1>       call   disk_read
3364 0000C9F5 5B                    <1>       pop    ebx
3365 0000C9F6 720B                  <1>       jc     short load_DirBuff_error
3366                                <1>
3367                                <1> validate_DirBuff_and_return:
3368 0000C9F8 59                    <1>       pop    ecx ; Number of loaded sectors
3369 0000C9F9 C605[94880100]01      <1>       mov    byte [DirBuff_ValidData], 1
3370 0000CA00 31C0                  <1>       xor    eax, eax ; 0 = no error
3371 0000CA02 C3                    <1>       retn
3372                                <1>
3373                                <1> load_DirBuff_error:
3374 0000CA03 89C8                  <1>       mov    eax, ecx ; remaining sectors
3375 0000CA05 59                    <1>       pop    ecx ; sector count
3376 0000CA06 29C1                  <1>       sub    ecx, eax ; Number of loaded sectors
3377                                <1>       ; 15/10/2016 (15h -> 17)
3378 0000CA08 B811000000            <1>       mov    eax, 17 ; DRV NOT READY OR READ ERROR !
3379 0000CA0D F9                    <1>       stc
3380 0000CA0E C3                    <1>       retn
3381                                <1>
3382                                <1> load_FAT32_root_directory:
3383                                <1>       ; 02/02/2016 (TRDOS 386 =  TRDOS v2.0)
3384                                <1>       ;
3385                                <1>       ; INPUT ->
3386                                <1>       ;      ESI = Logical DOS Drive Description Table
3387                                <1>       ; OUTPUT ->
3388                                <1>       ;      cf = 1 -> Root directory could not be loaded
3389                                <1>       ;          EAX > 0 -> Error number
3390                                <1>       ;      cf = 0 -> EAX = 0
3391                                <1>       ;      ECX = Directory buffer size in sectors (CL)
3392                                <1>       ;      EBX = Directory buffer address
3393                                <1>       ;      NOTE: DirBuffer_Size is in bytes ! (word)
3394                                <1>       ;
3395                                <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
3396                                <1>
3397                                <1>
3398 0000CA0F 8A1E                  <1>       mov    bl, [esi+LD_Name]
3399 0000CA11 8A7E03                <1>       mov    bh, [esi+LD_FATType]
3400                                <1>
3401                                <1>       ;mov   [DirBuff_DRV], bl
3402                                <1>       ;mov   [DirBuff_FATType], bh
3403 0000CA14 66891D[92880100]      <1>       mov    [DirBuff_DRV], bx
3404                                <1>
3405                                <1> load_FAT32_root_dir0:
3406 0000CA1B 8B4632                <1>       mov    eax, [esi+LD_BPB+FAT32_RootFClust]
3407 0000CA1E EB0C                  <1>       jmp    short load_FAT_sub_dir0
3408                                <1>
3409                                <1> load_FAT_sub_directory:
3410                                <1>       ; 01/02/2016 (TRDOS 386 =  TRDOS v2.0)
3411                                <1>       ; 05/07/2011
3412                                <1>       ; 23/08/2009
3413                                <1>       ;
3414                                <1>       ; INPUT ->
3415                                <1>       ;      ESI = Logical DOS Drive Description Table
3416                                <1>       ;      EAX = Cluster Number
3417                                <1>       ; OUTPUT ->
3418                                <1>       ;      cf = 1 -> Sub directory could not be loaded
3419                                <1>       ;          EAX > 0 -> Error number
3420                                <1>       ;      cf = 0 -> EAX = 0
3421                                <1>       ;      ECX = Directory buffer size in sectors (CL)
3422                                <1>       ;      EBX = Directory buffer address
3423                                <1>       ;
3424                                <1>       ;      NOTE: DirBuffer_Size is in bytes ! (word)
3425                                <1>       ;
3426                                <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
3427                                <1>
3428 0000CA20 8A1E                  <1>       mov    bl, [esi+LD_Name]
3429 0000CA22 8A7E03                <1>       mov    bh, [esi+LD_FATType]
3430                                <1>
3431                                <1>       ;mov   [DirBuff_DRV], bl
3432                                <1>       ;mov   [DirBuff_FATType], bh
3433 0000CA25 66891D[92880100]      <1>       mov    [DirBuff_DRV], bx
3434                                <1>
3435                                <1> load_FAT_sub_dir0:
3436 0000CA2C 0FB64E13              <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
3437                                <1>
3438 0000CA30 882D[94880100]        <1>       mov    [DirBuff_ValidData], ch ; 0
3439 0000CA36 A3[99880100]          <1>       mov    [DirBuff_Cluster], eax
3440                                <1>
3441 0000CA3B 0FB74611              <1>       movzx  eax, word [esi+LD_BPB+BytesPerSec]
3442 0000CA3F F7E1                  <1>       mul    ecx
3443 0000CA41 C1E805                <1>       shr    eax, 5 ; directory entry count (dir size / 32)
3444 0000CA44 6648                  <1>       dec    ax ; last entry
3445 0000CA46 66A3[97880100]        <1>       mov    [DirBuff_LastEntry], ax
3446                                <1>
3447 0000CA4C A1[99880100]          <1>       mov    eax, [DirBuff_Cluster]
3448 0000CA51 83E802                <1>       sub    eax, 2
3449 0000CA54 F7E1                  <1>       mul    ecx
3450 0000CA56 034668                <1>       add    eax, [esi+LD_DATABegin]
3451                                <1>       ; ecx = sector per cluster (dir buffer size = 32 sectors)
3452 0000CA59 EB8E                  <1>       jmp    short read_directory
3453                                <1>
3454                                <1> ; DRV_FS.ASM
3455                                <1>
3456                                <1> load_current_FS_directory:
3457 0000CA5B C3                    <1>       retn
3458                                <1> load_FS_root_directory:
3459 0000CA5C C3                    <1>       retn
3460                                <1> load_FS_sub_directory:
3461 0000CA5D C3                    <1>       retn
```

```
3462                                    <1>
3463                                    <1> read_cluster:
3464                                    <1>         ; 15/10/2016
3465                                    <1>         ; 18/03/2016
3466                                    <1>         ; 16/03/2016
3467                                    <1>         ; 17/02/2016
3468                                    <1>         ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
3469                                    <1>         ;
3470                                    <1>         ; INPUT ->
3471                                    <1>         ;       EAX = Cluster Number (Sector index for SINGLIX FS)
3472                                    <1>         ;       ESI = Logical DOS Drive Description Table address
3473                                    <1>         ;       EBX = Cluster (File R/W) Buffer address (max. 64KB)
3474                                    <1>         ;       Only for SINGLIX FS:
3475                                    <1>         ;       EDX = File Number (The 1st FDT address)
3476                                    <1>         ; OUTPUT ->
3477                                    <1>         ;       cf = 1 -> Cluster can not be loaded at the buffer
3478                                    <1>         ;           EAX > 0 -> Error number
3479                                    <1>         ;       cf = 0 -> Cluster has been loaded at the buffer
3480                                    <1>         ;
3481                                    <1>         ; (Modified registers: EAX, ECX, EBX, EDX)
3482                                    <1>
3483 0000CA5E 0FB64E13                 <1>         movzx  ecx, byte [esi+LD_BPB+BPB_SecPerClust]
3484                                    <1>         ; CL = 1 = [esi+LD_FS_Reserved2] ; SectPerClust for Singlix FS
3485                                    <1>
3486                                    <1> read_file_sectors: ; 16/03/2016
3487 0000CA62 807E0300                 <1>         cmp    byte [esi+LD_FATType], 0
3488 0000CA66 761C                     <1>         jna    short read_fs_cluster
3489                                    <1>
3490                                    <1> read_fat_file_sectors: ; 18/03/2016
3491 0000CA68 83E802                   <1>         sub    eax, 2 ; Beginning cluster number is always 2
3492 0000CA6B 0FB65613                 <1>         movzx  edx, byte [esi+LD_BPB+BPB_SecPerClust] ; 18/03/2016
3493 0000CA6F F7E2                     <1>         mul    edx
3494 0000CA71 034668                   <1>         add    eax, [esi+LD_DATABegin] ; absolute address of the cluster
3495                                    <1>
3496                                    <1>         ; EAX = Disk sector address
3497                                    <1>         ; ECX = Sector count
3498                                    <1>         ; EBX = Buffer address
3499                                    <1>         ; (EDX = 0)
3500                                    <1>         ; ESI = Logical DOS drive description table address
3501                                    <1>
3502 0000CA74 E8225B0000               <1>         call   disk_read
3503 0000CA79 7306                     <1>         jnc    short rclust_retn
3504                                    <1>
3505                                    <1>         ; 15/10/2016 (15h -> 17)
3506 0000CA7B B811000000               <1>         mov    eax, 17 ; Drive not ready or read error !
3507 0000CA80 C3                       <1>         retn
3508                                    <1>
3509                                    <1> rclust_retn:
3510 0000CA81 29C0                     <1>         sub    eax, eax ; 0
3511 0000CA83 C3                       <1>         retn
3512                                    <1>
3513                                    <1> read_fs_cluster:
3514                                    <1>         ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
3515                                    <1>         ; Singlix FS
3516                                    <1>
3517                                    <1>         ; EAX = Cluster number is sector index number of the file (eax)
3518                                    <1>
3519                                    <1>         ; EDX = File number is the first File Descriptor Table address
3520                                    <1>         ;       of the file. (Absolute address of the FDT).
3521                                    <1>
3522                                    <1>         ; eax = sector index (0 for the first sector)
3523                                    <1>         ; edx = FDT0 address
3524                                    <1>               ; 64 KB buffer = 128 sectors (limit)
3525 0000CA84 B980000000               <1>         mov    ecx, 128 ; maximum count of sectors (before eof)
3526 0000CA89 E801000000               <1>         call   read_fs_sectors
3527 0000CA8E C3                       <1>         retn
3528                                    <1>
3529                                    <1> read_fs_sectors:
3530                                    <1>         ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
3531 0000CA8F F9                       <1>         stc
3532 0000CA90 C3                       <1>         retn
3533                                    <1>
3534                                    <1> get_first_free_cluster:
3535                                    <1>         ; 02/03/2016
3536                                    <1>         ; 21/02/2016 (TRDOS 386 =  TRDOS v2.0)
3537                                    <1>         ; 26/10/2010 (DRV_FAT.ASM, 'proc_get_first_free_cluster')
3538                                    <1>         ; 10/07/2010
3539                                    <1>         ; INPUT ->
3540                                    <1>         ;       ESI = Logical DOS Drive Description Table address
3541                                    <1>         ; OUTPUT ->
3542                                    <1>         ;       cf = 1 -> Error code in AL (EAX)
3543                                    <1>         ;       cf = 0 ->
3544                                    <1>         ;         EAX = Cluster number
3545                                    <1>         ;         If EAX = FFFFFFFFh -> no free space
3546                                    <1>         ;       If the drive has FAT32 fs:
3547                                    <1>         ;         EBX = FAT32 FSI sector buffer address (if > 0)
3548                                    <1>
3549 0000CA91 8B4678                   <1>         mov    eax, [esi+LD_Clusters]
3550 0000CA94 40                       <1>         inc    eax ; add eax, 1
3551 0000CA95 A3[1C8B0100]             <1>         mov    [gffc_last_free_cluster], eax
3552                                    <1>
3553 0000CA9A 31DB                     <1>         xor    ebx, ebx ; 0 ; 02/03/2016
3554                                    <1>
3555 0000CA9C 807E0302                 <1>         cmp    byte [esi+LD_FATType], 2
3556 0000CAA0 760E                     <1>         jna    short loc_gffc_get_first_fat_free_cluster0
3557                                    <1>
3558                                    <1> loc_gffc_get_first_fat32_free_cluster:
3559                                    <1>         ; 02/03/2016
3560 0000CAA2 E844060000               <1>         call   get_fat32_fsinfo_sector_parms
3561 0000CAA7 7207                     <1>         jc     short loc_gffc_get_first_fat_free_cluster0
3562                                    <1>
3563                                    <1> loc_gffc_check_fsinfo_parms:
3564                                    <1>         ;;mov  ebx, DOSBootSectorBuff
3565                                    <1>         ;cmp   dword [ebx], 41615252h
3566                                    <1>         ;jne   short loc_gffc_fat32_fsinfo_err
```

```
3567                            <1>       ;cmp   dword [ebx+484], 61417272h
3568                            <1>       ;jne   short loc_gffc_fat32_fsinfo_err
3569                            <1>       ;mov   eax, [ebx+492] ; FSI_Next_Free
3570                            <1>       ;EAX = First free cluster
3571                            <1>       ;(from FAT32 FSInfo sector)
3572 0000CAA9 89D0              <1>       mov    eax, edx ; FSI_Next_Free (First Free Cluster)
3573 0000CAAB 83F8FF            <1>       cmp    eax, 0FFFFFFFFh ; invalid (unknown) !
3574 0000CAAE 7205              <1>       jb     short loc_gffc_get_first_fat_free_cluster1
3575                            <1>
3576                            <1>       ; Start from the 1st cluster of the FAT(32) file system
3577                            <1> loc_gffc_get_first_fat_free_cluster0:
3578 0000CAB0 B802000000        <1>       mov    eax, 2
3579                            <1>       ;xor   edx, edx
3580                            <1>
3581                            <1> loc_gffc_get_first_fat_free_cluster1:
3582 0000CAB5 53                <1>       push   ebx ; 02/03/2016
3583                            <1>
3584                            <1> loc_gffc_get_first_fat_free_cluster2:
3585 0000CAB6 A3[188B0100]      <1>       mov    [gffc_first_free_cluster], eax
3586 0000CABB A3[148B0100]      <1>       mov    [gffc_next_free_cluster], eax
3587                            <1>
3588                            <1>       ; EBX = FAT32 FSINFO sector buffer address
3589                            <1>       ; (EBX = 0, if the drive has not got FAT32 fs or
3590                            <1>       ; FAT32 FSINFO sector buffer is invalid.)
3591                            <1>
3592                            <1> loc_gffc_get_first_fat_free_cluster3:
3593 0000CAC0 E875FDFFFF        <1>       call   get_next_cluster
3594 0000CAC5 7307              <1>       jnc    short loc_gffc_get_first_fat_free_cluster4
3595 0000CAC7 09C0              <1>       or     eax, eax
3596 0000CAC9 740B              <1>       jz     short loc_gffc_first_free_fat_cluster_next
3597 0000CACB 5B                <1>       pop    ebx ; 02/03/2016
3598 0000CACC F5                <1>       cmc    ; stc
3599 0000CACD C3                <1>       retn
3600                            <1>
3601                            <1> loc_gffc_get_first_fat_free_cluster4:
3602 0000CACE 21C0              <1>       and    eax, eax ; next cluster value
3603 0000CAD0 7504              <1>       jnz    short loc_gffc_first_free_fat_cluster_next
3604 0000CAD2 89C8              <1>       mov    eax, ecx ; current (previous cluster) value
3605 0000CAD4 EB22              <1>       jmp    short loc_gffc_check_for_set
3606                            <1>
3607                            <1> loc_gffc_first_free_fat_cluster_next:
3608 0000CAD6 A1[148B0100]      <1>       mov    eax, [gffc_next_free_cluster]
3609 0000CADB 3B05[1C8B0100]    <1>       cmp    eax, [gffc_last_free_cluster]
3610 0000CAE1 7308              <1>       jnb    short retn_stc_from_get_first_free_cluster
3611                            <1> pass_gffc_last_cluster_eax_check:
3612 0000CAE3 40                <1>       inc    eax ; add eax, 1
3613 0000CAE4 A3[148B0100]      <1>       mov    [gffc_next_free_cluster], eax
3614 0000CAE9 EBD5              <1>       jmp    short loc_gffc_get_first_fat_free_cluster3
3615                            <1>
3616                            <1> retn_stc_from_get_first_free_cluster:
3617 0000CAEB A1[188B0100]      <1>       mov    eax, [gffc_first_free_cluster]
3618 0000CAF0 83F802            <1>       cmp    eax, 2
3619 0000CAF3 7709              <1>       ja     short loc_gffc_check_previous_clusters
3620 0000CAF5 29C0              <1>       sub    eax, eax
3621 0000CAF7 48                <1>       dec    eax ; FFFFFFFFh
3622                            <1>
3623                            <1> loc_gffc_check_for_set:
3624                            <1>       ; 02/03/2016
3625 0000CAF8 5B                <1>       pop    ebx
3626                            <1>
3627                            <1>       ; EBX = FAT32 FSINFO sector buffer address
3628                            <1>       ; (EBX = 0, if the drive has not got FAT32 fs or
3629                            <1>       ; FAT32 FSINFO sector buffer is invalid.)
3630                            <1>
3631 0000CAF9 09DB              <1>       or     ebx, ebx
3632 0000CAFB 750E              <1>       jnz    short loc_gffc_set_ffree_fat32_cluster
3633                            <1>
3634                            <1>       ;cmp   byte [esi+LD_FATType], 3
3635                            <1>       ;jnb   short loc_gffc_set_ffree_fat32_cluster
3636                            <1>
3637                            <1>       ;xor   ebx, ebx ; 0
3638                            <1>
3639                            <1> loc_gffc_retn:
3640 0000CAFD C3                <1>       retn
3641                            <1>
3642                            <1> loc_gffc_check_previous_clusters:
3643 0000CAFE 48                <1>       dec    eax ; sub eax, 1
3644 0000CAFF A3[1C8B0100]      <1>       mov    [gffc_last_free_cluster], eax
3645 0000CB04 B802000000        <1>       mov    eax, 2
3646                            <1>       ;xor   edx, edx
3647 0000CB09 EBAB              <1>       jmp    short loc_gffc_get_first_fat_free_cluster2
3648                            <1>
3649                            <1> loc_gffc_set_ffree_fat32_cluster:
3650                            <1>       ;call  set_first_free_cluster
3651                            <1>       ;retn
3652                            <1>       ;jmp   short set_first_free_cluster
3653                            <1>
3654                            <1> set_first_free_cluster:
3655                            <1>       ; 15/10/2016
3656                            <1>       ; 23/03/2016
3657                            <1>       ; 02/03/2016
3658                            <1>       ; 29/02/2016
3659                            <1>       ; 26/02/2016
3660                            <1>       ; 21/02/2016 (TRDOS 386 =  TRDOS v2.0)
3661                            <1>       ; 21/08/2011 (DRV_FAT.ASM, 'proc_set_first_free_cluster')
3662                            <1>       ; 11/07/2010
3663                            <1>       ; INPUT ->
3664                            <1>       ;      ESI = Logical DOS Drive Description Table address
3665                            <1>       ;      EAX = First free cluster
3666                            <1>       ;      EBX = FSINFO sector buffer address
3667                            <1>       ;      ;;If EBX > 0, it is FSINFO sector buffer address
3668                            <1>       ;      ;;EBX = 0, if FSINFO sector is not loaded
3669                            <1>       ; OUTPUT->
3670                            <1>       ;      ESI = Logical DOS Drive Description Table address
3671                            <1>       ;      If EBX > 0, it is FSINFO sector buffer address
```

```
3672                                    <1>       ;     EBX = 0, if FSINFO sector could not be loaded
3673                                    <1>       ;     CF = 1 -> Error code in AL (EAX)
3674                                    <1>       ;     CF = 0 -> first free cluster is successfully updated
3675                                    <1>
3676                                    <1>       ;cmp   byte [esi+LD_FATType], 3
3677                                    <1>       ;jb    short loc_sffc_invalid_drive
3678                                    <1>
3679                                    <1>       ; Save First Free Cluster value for 'update_cluster'
3680 0000CB0B 89463E                   <1>       mov    [esi+LD_BPB+BPB_Reserved+4], eax ; First free Cluster
3681                                    <1>
3682                                    <1>       ;or    ebx, ebx
3683                                    <1>       ;jnz   short loc_sffc_read_fsinfo_sector
3684                                    <1>
3685 0000CB0E 813B52526141             <1>       cmp     dword [ebx], 41615252h
3686 0000CB14 7540                     <1>       jne    short loc_sffc_read_fsinfo_sector
3687 0000CB16 81BBE4010000727241-      <1>       cmp    dword [ebx+484], 61417272h
3687 0000CB1F 61                       <1>
3688 0000CB20 7534                     <1>       jne    short loc_sffc_read_fsinfo_sector
3689                                    <1>
3690 0000CB22 3B83EC010000             <1>       cmp    eax, [ebx+492]  ; FSI_Next_Free
3691 0000CB28 741F                     <1>       je     short loc_sffc_retn
3692                                    <1>
3693                                    <1> loc_sffc_write_fsinfo_sector:
3694                                    <1>       ; EBX = FSINFO sector buffer
3695                                    <1>       ; [CFS_FAT32FSINFOSEC] is set in 'get_fat32_fsinfo_sector_parms'
3696 0000CB2A 8983EC010000             <1>       mov    [ebx+492], eax
3697 0000CB30 A1[2C8B0100]             <1>       mov    eax, [CFS_FAT32FSINFOSEC]
3698 0000CB35 B901000000               <1>       mov    ecx, 1
3699 0000CB3A 53                       <1>       push   ebx
3700 0000CB3B E84C5A0000               <1>       call   disk_write
3701 0000CB40 7208                     <1>       jc      short loc_sffc_read_fsinfo_sector_err1
3702 0000CB42 5B                       <1>       pop    ebx
3703                                    <1>
3704 0000CB43 8B83EC010000             <1>       mov    eax, [ebx+492] ; First (Next) Free Cluster
3705                                    <1>
3706                                    <1> loc_sffc_retn:
3707 0000CB49 C3                       <1>       retn
3708                                    <1>
3709                                    <1> ;loc_sffc_invalid_drive:
3710                                    <1> ;     mov    eax, 0Fh ; MSDOS Error : Invalid drive
3711                                    <1> ;     push   edx
3712                                    <1>
3713                                    <1> loc_sffc_read_fsinfo_sector_err1:
3714 0000CB4A BB00000000               <1>       mov    ebx, 0
3715                                    <1>       ; 15/10/2016 (1Dh -> 18)
3716                                    <1>       ; 23/03/2016 (1Dh)
3717 0000CB4F B812000000               <1>       mov    eax, 18 ; Drive not ready or write error
3718                                    <1>
3719                                    <1> loc_sffc_read_fsinfo_sector_err2:
3720 0000CB54 5A                       <1>       pop    edx
3721 0000CB55 C3                       <1>       retn
3722                                    <1>
3723                                    <1> loc_sffc_read_fsinfo_sector:
3724 0000CB56 50                       <1>       push   eax
3725                                    <1>
3726 0000CB57 E88F050000               <1>       call   get_fat32_fsinfo_sector_parms
3727 0000CB5C 72F6                     <1>       jc     short loc_sffc_read_fsinfo_sector_err2
3728                                    <1>
3729 0000CB5E 58                       <1>       pop    eax
3730                                    <1>       ; EDX = First (Next) Free Cluster value from FSINFO sector
3731                                    <1>       ; EAX = First Free Cluster value from 'get_next_cluster'
3732                                    <1>       ; (edx = old value)
3733 0000CB5F 39D0                     <1>       cmp    eax, edx ; First free Cluster (eax = new value)
3734 0000CB61 75C7                     <1>       jne    short loc_sffc_write_fsinfo_sector
3735                                    <1>
3736 0000CB63 C3                       <1>       retn
3737                                    <1>
3738                                    <1> update_cluster:
3739                                    <1>       ; 23/10/2016
3740                                    <1>       ; 23/03/2016
3741                                    <1>       ; 02/03/2016
3742                                    <1>       ; 01/03/2016
3743                                    <1>       ; 29/02/2016
3744                                    <1>       ; 27/02/2016
3745                                    <1>       ; 26/02/2016
3746                                    <1>       ; 22/02/2016 (TRDOS 386 =  TRDOS v2.0)
3747                                    <1>       ; 11/08/2011
3748                                    <1>       ; 09/02/2005
3749                                    <1>       ; INPUT ->
3750                                    <1>       ;     EAX = Cluster Number
3751                                    <1>       ;     ECX = New Cluster Value
3752                                    <1>       ;     ESI = Logical Dos Drive Parameters Table
3753                                    <1>       ;
3754                                    <1>       ;     /// dword [FAT_ClusterCounter] ///
3755                                    <1>       ;
3756                                    <1>       ; OUTPUT ->
3757                                    <1>       ;     cf = 0 -> No Error, EAX is valid
3758                                    <1>       ;     cf = 1 & EAX = 0 -> End Of Cluster Chain
3759                                    <1>       ;     cf = 1 & EAX > 0 -> Error
3760                                    <1>       ;           (ECX -> any value)
3761                                    <1>       ;     EAX = Next Cluster
3762                                    <1>       ;     ECX = New Cluster Value
3763                                    <1>       ;
3764                                    <1>       ;     /// [FAT_ClusterCounter] is updated,
3765                                    <1>       ;     /// decreased when a free cluster is assigned,
3766                                    <1>       ;     /// increased if an assigned cluster is freed.
3767                                    <1>       ;
3768                                    <1>       ;
3769                                    <1>       ; (Modified registers: EAX, EBX, -ECX-, EDX)
3770                                    <1>
3771 0000CB64 A3[7E880100]             <1>       mov    [FAT_CurrentCluster], eax
3772 0000CB69 890D[208B0100]           <1>       mov    [ClusterValue], ecx
3773                                    <1>
3774                                    <1> loc_update_cluster_check_fat_buffer:
3775 0000CB6F 8A1E                     <1>       mov    bl, [esi+LD_Name]
```

```
3776 0000CB71 381D[83880100]      <1>       cmp    [FAT_BuffDrvName], bl
3777 0000CB77 741A                <1>       je     short loc_update_cluster_check_fat_type
3778 0000CB79 803D[82880100]02    <1>       cmp    byte [FAT_BuffValidData], 2
3779 0000CB80 0F84C2000000        <1>       je     loc_uc_save_fat_buffer
3780                              <1>
3781                              <1> loc_uc_reset_fat_buffer_validation:
3782 0000CB86 C605[82880100]00    <1>       mov    byte [FAT_BuffValidData], 0
3783                              <1>
3784                              <1> loc_uc_check_fat_type_reset_drvname:
3785 0000CB8D 881D[83880100]      <1>       mov    [FAT_BuffDrvName], bl
3786                              <1>
3787                              <1> loc_update_cluster_check_fat_type:
3788 0000CB93 29D2                <1>       sub    edx, edx ; 26/02/2016
3789 0000CB95 8A5E03              <1>       mov    bl, [esi+LD_FATType]
3790 0000CB98 83F802              <1>       cmp    eax, 2
3791 0000CB9B 0F82BE000000        <1>       jb     update_cluster_inv_data
3792 0000CBA1 80FB02              <1>       cmp    bl, 2
3793 0000CBA4 0F877A010000        <1>       ja     update_fat32_cluster
3794                              <1>       ;cmp   bl, 1
3795                              <1>       ;jb    short update_cluster_inv_data
3796 0000CBAA 8B4E78              <1>       mov    ecx, [esi+LD_Clusters]
3797 0000CBAD 41                  <1>       inc    ecx
3798 0000CBAE 890D[8E880100]      <1>       mov    [LastCluster], ecx
3799 0000CBB4 39C8                <1>       cmp    eax, ecx ; dword [LastCluster]
3800 0000CBB6 0F87A6000000        <1>       ja     return_uc_fat_stc
3801                              <1>       ; TRDOS v1 has a FATal bug here !
3802                              <1>              ; or bl, bl ; cmp bl, 0
3803                              <1>              ; jz short update_fat12_cluster
3804                              <1>       ; !! It would destroy FAT12 floppy disk fs here !!
3805                              <1>       ; ('A:' disks of TRDOS v1 operating system project
3806                              <1>       ; had 'singlix fs', so, I could not differ this mistake
3807                              <1>       ; on a drive 'A:')
3808 0000CBBC 80FB01              <1>       cmp    bl, 1 ; correct comparison is this !
3809 0000CBBF 0F86A2000000        <1>       jna    update_fat12_cluster
3810                              <1>
3811                              <1> update_fat16_cluster:
3812                              <1> pass_uc_fat16_errc:
3813                              <1>       ;sub   edx, edx
3814 0000CBC5 BB00030000          <1>       mov    ebx, 300h ;768
3815 0000CBCA F7F3                <1>       div    ebx
3816                              <1>       ; EAX = Count of 3 FAT sectors
3817                              <1>       ; DX = Cluster offset in FAT buffer
3818 0000CBCC 6689D3              <1>       mov    bx, dx
3819 0000CBCF 66D1E3              <1>       shl    bx, 1 ; Multiply by 2
3820 0000CBD2 66BA0300            <1>       mov    dx, 3
3821 0000CBD6 F7E2                <1>       mul    edx
3822                              <1>       ; EAX = FAT Sector
3823                              <1>       ; EDX = 0
3824                              <1>       ; EBX = Byte offset in FAT buffer
3825 0000CBD8 8A0D[82880100]      <1>       mov    cl, [FAT_BuffValidData]
3826 0000CBDE 80F902              <1>       cmp    cl, 2
3827 0000CBE1 750A                <1>       jne    short loc_uc_check_fat16_buff_sector_load
3828                              <1>
3829                              <1> loc_uc_check_fat16_buff_sector_save:
3830 0000CBE3 3B05[86880100]      <1>       cmp    eax, [FAT_BuffSector]
3831 0000CBE9 755D                <1>       jne    short loc_uc_save_fat_buffer
3832 0000CBEB EB15                <1>       jmp    short loc_update_fat16_cell
3833                              <1>
3834                              <1> loc_uc_check_fat16_buff_sector_load:
3835 0000CBED 80F901              <1>       cmp    cl, 1 ; byte [FAT_BuffValidData]
3836 0000CBF0 0F85FB010000        <1>       jne    loc_uc_load_fat_sectors
3837 0000CBF6 3B05[86880100]      <1>       cmp    eax, [FAT_BuffSector]
3838 0000CBFC 0F85EF010000        <1>       jne    loc_uc_load_fat_sectors
3839                              <1>
3840                              <1> loc_update_fat16_cell:
3841                              <1> loc_update_fat16_buffer:
3842 0000CC02 81C3001C0900        <1>       add    ebx, FAT_Buffer ; 26/02/2016
3843                              <1>       ;movzx eax, word [ebx]
3844 0000CC08 668B03              <1>       mov    ax, [ebx]
3845                              <1>       ; 01/03/2016
3846 0000CC0B 89C2                <1>       mov    edx, eax ; old value of the cluster
3847 0000CC0D A3[7E880100]        <1>       mov    [FAT_CurrentCluster], eax
3848 0000CC12 8B0D[208B0100]      <1>       mov    ecx, [ClusterValue] ; 32 bits
3849 0000CC18 66890B              <1>       mov    [ebx], cx ; 16 bits !
3850                              <1>
3851 0000CC1B C605[82880100]02    <1>       mov    byte [FAT_BuffValidData], 2
3852                              <1>
3853 0000CC22 6683F802            <1>       cmp    ax, 2
3854 0000CC26 723A                <1>       jb     short return_uc_fat_stc
3855 0000CC28 3B05[8E880100]      <1>       cmp    eax, [LastCluster]
3856 0000CC2E 7732                <1>       ja     short return_uc_fat_stc
3857                              <1>
3858                              <1> loc_fat_buffer_updated:
3859                              <1>       ; 01/03/2016
3860 0000CC30 F8                  <1>       clc
3861                              <1> loc_fat_buffer_stc_1:
3862 0000CC31 9C                  <1>       pushf
3863 0000CC32 21C9                <1>       and    ecx, ecx
3864 0000CC34 7506                <1>       jnz    short loc_fat_buffer_updated_1
3865                              <1>
3866                              <1>       ; 01/03/2016
3867                              <1>       ; new value of the cluster = 0 (free)
3868                              <1>       ; increase free(d) cluster count
3869 0000CC36 FF05[8A880100]      <1>       inc    dword [FAT_ClusterCounter]
3870                              <1>
3871                              <1> loc_fat_buffer_updated_1: ; new value of the cluster > 0
3872 0000CC3C 09D2                <1>       or     edx, edx ; 02/03/2016
3873 0000CC3E 7506                <1>       jnz    short loc_fat_buffer_updated_2
3874                              <1>       ; old value of the cluster = 0 (it was free cluster)
3875                              <1>       ; decrease free(d) cluster count
3876 0000CC40 FF0D[8A880100]      <1>       dec    dword [FAT_ClusterCounter] ; it may be negative number
3877                              <1>
3878                              <1> loc_fat_buffer_updated_2:
3879 0000CC46 9D                  <1>       popf
3880 0000CC47 C3                  <1>       retn
```

```
                                <1>
                                <1> loc_uc_save_fat_buffer:
                                <1>       ; byte [FAT_BuffValidData] = 2
0000CC48 E8D4010000             <1>       call   save_fat_buffer
0000CC4D 0F8297010000           <1>       jc     loc_fat_sectors_rw_error2
                                <1>       ;mov   byte [FAT_BuffValidData], 1
0000CC53 A1[7E880100]           <1>       mov    eax, [FAT_CurrentCluster]
                                <1>       ;mov   ecx, [ClusterValue]
                                <1>       ;jmp   short loc_update_cluster_check_fat_buffer
0000CC58 8A1E                   <1>       mov    bl, [esi+LD_Name] ; 01/03/2016
0000CC5A E927FFFFFF             <1>       jmp    loc_uc_reset_fat_buffer_validation
                                <1>
                                <1> update_cluster_inv_data:
                                <1>       ;mov   eax, 0Dh
0000CC5F B00D                   <1>       mov    al, 0Dh  ; Invalid Data
0000CC61 C3                     <1>       retn
                                <1>
                                <1> return_uc_fat_stc:
                                <1>       ; 01/03/2016
0000CC62 31C0                   <1>       xor    eax, eax
0000CC64 F9                     <1>       stc
0000CC65 EBCA                   <1>       jmp    short loc_fat_buffer_stc_1
                                <1>
                                <1> update_fat12_cluster:
                                <1> pass_uc_fat12_errc:
                                <1>       ;sub   edx, edx
0000CC67 BB00040000             <1>       mov    ebx, 400h ;1024
0000CC6C F7F3                   <1>       div    ebx
                                <1>       ; EAX = Count of 3 FAT sectors
                                <1>       ; DX = Cluster offset in FAT buffer
0000CC6E 66B90300               <1>       mov    cx, 3
0000CC72 6689C3                 <1>       mov    bx, ax
0000CC75 6689C8                 <1>       mov    ax, cx ; 3
0000CC78 66F7E2                 <1>       mul    dx     ; Multiply by 3
0000CC7B 66D1E8                 <1>       shr    ax, 1  ; Divide by 2
0000CC7E 6693                   <1>       xchg   bx, ax
                                <1>       ; EAX = Count of 3 FAT sectors
                                <1>       ; EBX = Byte Offset in FAT buffer
0000CC80 66F7E1                 <1>       mul    cx  ; 3 * AX
                                <1>       ; EAX = FAT Beginning Sector
                                <1>       ; EDX = 0
0000CC83 8A0D[82880100]         <1>       mov    cl, [FAT_BuffValidData]
                                <1>       ; TRDOS v1 has a FATal bug here !
                                <1>       ; (it does not have 'cmp cl, 2' instruction here !
                                <1>       ;  while 'jne' is existing !)
0000CC89 80F902                 <1>       cmp    cl, 2 ; 2 = dirty buffer (must be written to disk)
0000CC8C 750A                   <1>       jne    short loc_uc_check_fat12_buff_sector_load
                                <1>
                                <1> loc_uc_check_fat12_buff_sector_save:
0000CC8E 3B05[86880100]         <1>       cmp    eax, [FAT_BuffSector]
0000CC94 75B2                   <1>       jne    short loc_uc_save_fat_buffer
0000CC96 EB15                   <1>       jmp    short loc_update_fat12_cell
                                <1>
                                <1> loc_uc_check_fat12_buff_sector_load:
0000CC98 80F901                 <1>       cmp    cl, 1 ; byte ptr [FAT_BuffValidData]
0000CC9B 0F8550010000           <1>       jne    loc_uc_load_fat_sectors
0000CCA1 3B05[86880100]         <1>       cmp    eax, [FAT_BuffSector]
0000CCA7 0F8544010000           <1>       jne    loc_uc_load_fat_sectors
                                <1>
                                <1> loc_update_fat12_cell:
0000CCAD 81C3001C0900           <1>       add    ebx, FAT_Buffer ; 26/02/2016
0000CCB3 668B0D[7E880100]       <1>       mov    cx, [FAT_CurrentCluster]
0000CCBA 66D1E9                 <1>       shr    cx, 1
0000CCBD 668B03                 <1>       mov    ax, [ebx]
0000CCC0 6689C2                 <1>       mov    dx, ax
0000CCC3 7344                   <1>       jnc    short uc_fat12_nc_even
                                <1>
0000CCC5 6683E00F               <1>       and    ax, 0Fh
0000CCC9 8B0D[208B0100]         <1>       mov    ecx, [ClusterValue] ; 32 bits
0000CCCF 66C1E104               <1>       shl    cx, 4
0000CCD3 6609C1                 <1>       or     cx, ax
0000CCD6 6689D0                 <1>       mov    ax, dx
0000CCD9 66890B                 <1>       mov    [ebx], cx  ; 16 bits !
0000CCDC 66C1E804               <1>       shr    ax, 4 ; al(bit4..7)+ah(bit0..7)
                                <1>
                                <1> update_fat12_buffer:
0000CCE0 A3[7E880100]           <1>       mov    [FAT_CurrentCluster], eax
0000CCE5 89C2                   <1>       mov    edx, eax ; 01/03/2016
0000CCE7 C605[82880100]02       <1>       mov    byte [FAT_BuffValidData], 2
0000CCEE 6683F802               <1>       cmp    ax, 2
0000CCF2 0F826AFFFFFF           <1>       jb     return_uc_fat_stc
0000CCF8 3B05[8E880100]         <1>       cmp    eax, [LastCluster]
0000CCFE 0F875EFFFFFF           <1>       ja     return_uc_fat_stc
0000CD04 E927FFFFFF             <1>       jmp    loc_fat_buffer_updated
                                <1>
                                <1> uc_fat12_nc_even:
0000CD09 662500F0               <1>       and    ax, 0F000h
0000CD0D 8B0D[208B0100]         <1>       mov    ecx, [ClusterValue] ; 32 bits
0000CD13 80E50F                 <1>       and    ch, 0Fh
0000CD16 6609C1                 <1>       or     cx, ax
0000CD19 6689D0                 <1>       mov    ax, dx
0000CD1C 66890B                 <1>       mov    [ebx], cx ; 16 bits !
0000CD1F 80E40F                 <1>       and    ah, 0Fh ; al(bit0..7)+ah(bit0..3)
0000CD22 EBBC                   <1>       jmp    short update_fat12_buffer
                                <1>
                                <1> update_fat32_cluster:
0000CD24 8B4E78                 <1>       mov    ecx, [esi+LD_Clusters]
0000CD27 41                     <1>       inc    ecx
0000CD28 890D[8E880100]         <1>       mov    [LastCluster], ecx
                                <1>
0000CD2E 39C8                   <1>       cmp    eax, ecx
0000CD30 0F872CFFFFFF           <1>       ja     return_uc_fat_stc
                                <1>
                                <1> pass_uc_fat32_errc:
                                <1>       ;sub   edx, edx
```

```
3986 0000CD36 BB80010000          <1>       mov    ebx, 180h ;384
3987 0000CD3B F7F3                <1>       div    ebx
3988                              <1>       ; EAX = Count of 3 FAT sectors
3989                              <1>       ; DX = Cluster offset in FAT buffer
3990 0000CD3D 89D3                <1>       mov    ebx, edx
3991 0000CD3F C1E302              <1>       shl    ebx, 2 ; Multiply by 4
3992 0000CD42 BA03000000          <1>       mov    edx, 3
3993 0000CD47 F7E2                <1>       mul    edx
3994                              <1>       ; EBX = Cluster Offset in FAT buffer
3995                              <1>       ; EAX = FAT Sector
3996                              <1>       ; EDX = 0
3997 0000CD49 8A0D[82880100]      <1>       mov    cl, [FAT_BuffValidData]
3998 0000CD4F 80F902              <1>       cmp    cl, 2
3999 0000CD52 750E                <1>       jne    short loc_uc_check_fat32_buff_sector_load
4000                              <1>
4001                              <1> loc_uc_check_fat32_buff_sector_save:
4002 0000CD54 3B05[86880100]      <1>       cmp    eax, [FAT_BuffSector]
4003 0000CD5A 0F85E8FEFFFF        <1>       jne    loc_uc_save_fat_buffer
4004 0000CD60 EB11                <1>       jmp    short loc_update_fat32_cell
4005                              <1>
4006                              <1> loc_uc_check_fat32_buff_sector_load:
4007 0000CD62 80F901              <1>       cmp    cl, 1 ; byte [FAT_BuffValidData]
4008 0000CD65 0F8586000000        <1>       jne    loc_uc_load_fat_sectors
4009 0000CD6B 3B05[86880100]      <1>       cmp    eax, [FAT_BuffSector]
4010 0000CD71 757E                <1>       jne    loc_uc_load_fat_sectors
4011                              <1>
4012                              <1> loc_update_fat32_cell:
4013                              <1> loc_update_fat32_buffer:
4014 0000CD73 81C3001C0900        <1>       add    ebx, FAT_Buffer ; 26/02/2016
4015 0000CD79 8B03                <1>       mov    eax, [ebx]
4016 0000CD7B 25FFFFFF0F          <1>       and    eax, 0FFFFFFFh ; 28 bit cluster value
4017                              <1>
4018 0000CD80 8B15[7E880100]      <1>       mov    edx, [FAT_CurrentCluster] ; 01/03/2016
4019                              <1>
4020 0000CD86 A3[7E880100]        <1>       mov    [FAT_CurrentCluster], eax
4021 0000CD8B 8B0D[208B0100]      <1>       mov    ecx, [ClusterValue]
4022 0000CD91 890B                <1>       mov    [ebx], ecx ; 29/02/2016
4023                              <1>
4024 0000CD93 C605[82880100]02    <1>       mov    byte [FAT_BuffValidData], 2
4025                              <1>
4026                              <1>       ; 01/03/2016
4027 0000CD9A 21C0                <1>       and    eax, eax ; was it free cluster ?
4028 0000CD9C 7514                <1>       jnz    short loc_upd_fat32_c0
4029                              <1>
4030                              <1>       ;or    ecx, ecx ; it will be left free ?!
4031                              <1>       ;jz    short loc_upd_fat32_c3
4032                              <1>
4033 0000CD9E 3B563E              <1>       cmp    edx, [esi+LD_BPB+BPB_Reserved+4] ; First free cluster
4034 0000CDA1 7520                <1>       jne    short loc_upd_fat32_c3
4035                              <1>
4036 0000CDA3 3B15[8E880100]      <1>       cmp    edx, [LastCluster]
4037 0000CDA9 7207                <1>       jb     short loc_upd_fat32_c0
4038                              <1>
4039 0000CDAB BA02000000          <1>       mov    edx, 2 ; rewind !
4040 0000CDB0 EB0E                <1>       jmp    short loc_upd_fat32_c2
4041                              <1>
4042                              <1> loc_upd_fat32_c0:
4043 0000CDB2 FF463E              <1>       inc    dword [esi+LD_BPB+BPB_Reserved+4] ; set it to next cluster
4044 0000CDB5 EB0C                <1>       jmp    short loc_upd_fat32_c3
4045                              <1>
4046                              <1> loc_upd_fat32_c1:
4047 0000CDB7 09C9                <1>       or     ecx, ecx ; will it be free cluster ?
4048 0000CDB9 7508                <1>       jnz    short loc_upd_fat32_c3
4049                              <1>
4050 0000CDBB 3B563E              <1>       cmp    edx, [esi+LD_BPB+BPB_Reserved+4] ; First free cluster
4051 0000CDBE 7303                <1>       jnb    short loc_upd_fat32_c3
4052                              <1>
4053                              <1> loc_upd_fat32_c2:
4054 0000CDC0 89563E              <1>       mov    [esi+LD_BPB+BPB_Reserved+4], edx
4055                              <1>
4056                              <1> loc_upd_fat32_c3:
4057 0000CDC3 89C2                <1>       mov    edx, eax
4058                              <1>
4059                              <1> loc_upd_fat32_c4:
4060 0000CDC5 83F802              <1>       cmp    eax, 2
4061 0000CDC8 0F8294FEFFFF        <1>       jb     return_uc_fat_stc
4062                              <1>
4063                              <1> pass_uc_fat32_c_zero_check_2:
4064 0000CDCE 3B05[8E880100]      <1>       cmp    eax, [LastCluster]
4065 0000CDD4 0F8788FEFFFF        <1>       ja     return_uc_fat_stc
4066                              <1>
4067 0000CDDA E951FEFFFF          <1>       jmp    loc_fat_buffer_updated
4068                              <1>
4069                              <1> loc_fat_sectors_rw_error1:
4070                              <1>       ;mov   byte [FAT_BuffValidData], 0
4071                              <1>       ; 23/10/2016 (15h -> 17)
4072                              <1>       ; 23/03/2016
4073 0000CDDF B811000000          <1>       mov    eax, 17 ; Drive not ready or read error
4074 0000CDE4 8825[82880100]      <1>       mov    [FAT_BuffValidData], ah ; 0
4075                              <1>
4076                              <1> loc_fat_sectors_rw_error2:
4077                              <1>       ;mov   eax, error code
4078                              <1>       ;mov   edx, 0
4079 0000CDEA 8B0D[208B0100]      <1>       mov    ecx, [ClusterValue]
4080 0000CDF0 C3                  <1>       retn
4081                              <1>
4082                              <1> loc_uc_load_fat_sectors:
4083 0000CDF1 A3[86880100]        <1>       mov    [FAT_BuffSector], eax
4084                              <1>
4085                              <1> load_uc_fat_sectors_zero:
4086 0000CDF6 034660              <1>       add    eax, [esi+LD_FATBegin]
4087 0000CDF9 BB001C0900          <1>       mov    ebx, FAT_Buffer
4088 0000CDFE B903000000          <1>       mov    ecx, 3
4089 0000CE03 E893570000          <1>       call   disk_read
4090 0000CE08 72D5                <1>       jc     short loc_fat_sectors_rw_error1
```

```
4091                                <1>
4092 0000CE0A C605[82880100]01      <1>       mov    byte [FAT_BuffValidData], 1
4093 0000CE11 A1[7E880100]          <1>       mov    eax, [FAT_CurrentCluster]
4094 0000CE16 8B0D[208B0100]        <1>       mov    ecx, [ClusterValue]
4095 0000CE1C E972FDFFFF            <1>       jmp    loc_update_cluster_check_fat_type
4096                                <1>
4097                                <1> save_fat_buffer:
4098                                <1>       ; 15/10/2016
4099                                <1>       ; 01/03/2016
4100                                <1>       ; 22/02/2016 (TRDOS 386 =  TRDOS v2.0)
4101                                <1>       ; 11/08/2011
4102                                <1>       ; 09/02/2005
4103                                <1>       ; INPUT ->
4104                                <1>       ;      None
4105                                <1>       ; OUTPUT ->
4106                                <1>       ;      cf = 0 -> OK.
4107                                <1>       ;      cf = 1 -> error code in AL (EAX)
4108                                <1>       ;
4109                                <1>       ;      EBX = FAT_Buffer address
4110                                <1>       ;
4111                                <1>       ; (EAX, EDX, ECX will be modified)
4112                                <1>
4113                                <1>       ;cmp   byte [FAT_BuffValidData], 2
4114                                <1>       ;je    short loc_save_fat_buff
4115                                <1>
4116                                <1> ;loc_save_fat_buffer_retn:
4117                                <1> ;     xor    eax, eax
4118                                <1> ;     retn
4119                                <1>
4120                                <1> loc_save_fat_buff:
4121 0000CE21 31D2                 <1>       xor    edx, edx
4122 0000CE23 8A35[83880100]       <1>       mov    dh, [FAT_BuffDrvName]
4123 0000CE29 80FE41               <1>       cmp    dh, 'A'
4124 0000CE2C 722E                 <1>       jb     short loc_save_fat_buffer_inv_data_retn
4125 0000CE2E 80EE41               <1>       sub    dh, 'A'
4126 0000CE31 56                   <1>       push   esi ; *
4127 0000CE32 BE00010900           <1>         mov     esi, Logical_DOSDisks
4128 0000CE37 01D6                 <1>       add    esi, edx
4129                                <1>
4130 0000CE39 8A5603               <1>       mov    dl, [esi+LD_FATType]
4131 0000CE3C 20D2                 <1>       and    dl, dl
4132 0000CE3E 741B                 <1>       jz     short loc_save_fat_buffer_inv_data_pop_retn
4133                                <1>
4134 0000CE40 A1[86880100]         <1>       mov    eax, [FAT_BuffSector]
4135 0000CE45 80FA02               <1>       cmp    dl, 2
4136 0000CE48 770A                 <1>       ja     short loc_save_fat32_buff
4137                                <1>
4138                                <1> loc_save_fat_12_16_buff:
4139                                <1>       ; 01/03/2016
4140                                <1>       ; TRDOS v1 has a FATal bug here!
4141                                <1>       ; Correct code: mov dx, word ptr [FAT_BuffSector]+2
4142                                <1>       ; (DX:AX in TRDOS v1 -> EAX in TRDOS v2)
4143                                <1>       ;
4144 0000CE4A 0FB74E1C             <1>       movzx  ecx, word [esi+LD_BPB+FATSecs]
4145 0000CE4E 29C1                 <1>       sub    ecx, eax
4146                                <1>       ; TRDOS v1 has a bug here... ('pop esi' was forgotten!)
4147                                <1>       ;jna   short loc_save_fat_buffer_inv_data_retn ; wrong addr!
4148 0000CE50 7609                 <1>       jna    short loc_save_fat_buffer_inv_data_pop_retn ; correct addr.
4149 0000CE52 EB15                 <1>       jmp    short loc_save_fat_buffer_check_rs3
4150                                <1>
4151                                <1> loc_save_fat32_buff:
4152 0000CE54 8B4E2A               <1>       mov    ecx, [esi+LD_BPB+FAT32_FAT_Size]
4153 0000CE57 29C1                 <1>       sub    ecx, eax
4154 0000CE59 770E                 <1>       ja     short loc_save_fat_buffer_check_rs3
4155                                <1>
4156                                <1> loc_save_fat_buffer_inv_data_pop_retn:
4157 0000CE5B 5E                   <1>       pop    esi ; *
4158                                <1> loc_save_fat_buffer_inv_data_retn:
4159 0000CE5C B80D000000           <1>       mov    eax, 0Dh ; Invalid DATA
4160 0000CE61 C3                   <1>       retn
4161                                <1>
4162                                <1> loc_save_fat_buff_remain_sectors_3:
4163 0000CE62 B903000000           <1>       mov    ecx, 3
4164 0000CE67 EB05                 <1>       jmp    short loc_save_fat_buff_continue
4165                                <1>
4166                                <1> loc_save_fat_buffer_check_rs3:
4167 0000CE69 83F903               <1>       cmp    ecx, 3
4168 0000CE6C 77F4                 <1>       ja     short loc_save_fat_buff_remain_sectors_3
4169                                <1>
4170                                <1> loc_save_fat_buff_continue:
4171 0000CE6E BB001C0900           <1>       mov    ebx, FAT_Buffer
4172 0000CE73 034660               <1>       add    eax, [esi+LD_FATBegin]
4173 0000CE76 51                   <1>       push   ecx
4174 0000CE77 E810570000           <1>       call   disk_write
4175 0000CE7C 59                   <1>       pop    ecx
4176 0000CE7D 722B                 <1>       jc     short loc_save_FAT_buff_write_err
4177                                <1>
4178 0000CE7F 807E0302             <1>       cmp    byte [esi+LD_FATType], 2
4179 0000CE83 7605                 <1>       jna    short loc_calc_2nd_fat12_16_addr
4180                                <1>
4181                                <1> loc_calc_2nd_fat32_addr:
4182 0000CE85 8B462A               <1>       mov    eax, [esi+LD_BPB+FAT32_FAT_Size]
4183 0000CE88 EB04                 <1>       jmp    short loc_calc_2nd_fat_addr
4184                                <1>
4185                                <1> loc_calc_2nd_fat12_16_addr:
4186 0000CE8A 0FB7461C             <1>       movzx  eax, word [esi+LD_BPB+FATSecs]
4187                                <1>
4188                                <1> loc_calc_2nd_fat_addr:
4189 0000CE8E 034660               <1>       add    eax, [esi+LD_FATBegin]
4190 0000CE91 0305[86880100]       <1>       add    eax, [FAT_BuffSector]
4191 0000CE97 BB001C0900           <1>       mov    ebx, FAT_Buffer
4192                                <1>       ; ecx = 1 to 3
4193 0000CE9C E8EB560000           <1>       call   disk_write
4194 0000CEA1 7207                 <1>       jc     short loc_save_FAT_buff_write_err
4195                                <1>       ; Valid  buffer (1 = valid but do not save)
```

```
4196 0000CEA3 C605[82880100]01    <1>       mov    byte [FAT_BuffValidData], 1
4197                              <1>
4198                              <1> loc_save_FAT_buff_write_err:
4199 0000CEAA 5E                  <1>       pop    esi ; *
4200 0000CEAB BB001C0900          <1>       mov    ebx, FAT_Buffer
4201                              <1>       ; 15/10/2016 (1Dh -> 18)
4202                              <1>       ; 23/03/2016 (1Dh)
4203 0000CEB0 B812000000          <1>       mov    eax, 18 ; Drive not ready or write error
4204 0000CEB5 C3                  <1>       retn
4205                              <1>
4206                              <1> calculate_fat_freespace:
4207                              <1>       ; 23/03/2016
4208                              <1>       ; 02/03/2016
4209                              <1>       ; 01/03/2016
4210                              <1>       ; 29/02/2016
4211                              <1>       ; 22/02/2016 (TRDOS 386 =  TRDOS v2.0)
4212                              <1>       ; 30/04/2011
4213                              <1>       ; 03/04/2010
4214                              <1>       ; 2005
4215                              <1>       ; INPUT ->
4216                              <1>       ;      EAX = Cluster count to be added or subtracted
4217                              <1>       ;      If BH = FFh, ESI = TR-DOS Logical Drive Description Table
4218                              <1>       ;      If BH < FFh, BH = TR-DOS Logical Drive Number
4219                              <1>       ;      BL:
4220                              <1>       ;      0 = Calculate, 1 = Add, 2 = Subtract, 3 = Get (Not Set/Calc)
4221                              <1>       ; OUTPUT ->
4222                              <1>       ;      EAX = Free Space in sectors
4223                              <1>       ;      ESI = Logical Dos Drive Description Table address
4224                              <1>       ;      BH = Logical Dos Drive Number (same with input value of BH)
4225                              <1>       ;      BL = Type of operation (same with input value of BL)
4226                              <1>       ;      ECX = 0 -> valid
4227                              <1>       ;      ECX > 0 -> error or invalid
4228                              <1>       ;      If EAX = FFFFFFFFh, it is 're-calculation needed'
4229                              <1>       ;                              sign due to r/w error
4230                              <1>
4231 0000CEB6 66891D[268B0100]    <1>       mov    [CFS_OPType], bx
4232 0000CEBD A3[288B0100]        <1>       mov    [CFS_CC], eax
4233                              <1>
4234 0000CEC2 80FFFF              <1>       cmp    bh, 0FFh
4235 0000CEC5 740B                <1>       je     short pass_calculate_freespace_get_drive_dt_offset
4236                              <1>
4237                              <1> loc_calculate_freespace_get_drive_dt_offset:
4238 0000CEC7 31C0                <1>       xor    eax, eax
4239 0000CEC9 88FC                <1>       mov    ah, bh
4240 0000CECB BE00010900          <1>       mov    esi, Logical_DOSDisks
4241 0000CED0 01C6                <1>       add    esi, eax
4242                              <1>
4243                              <1> pass_calculate_freespace_get_drive_dt_offset:
4244 0000CED2 08DB                <1>       or     bl, bl
4245 0000CED4 7435                <1>       jz     short loc_reset_fcc
4246                              <1>
4247                              <1> loc_get_free_sectors:
4248 0000CED6 8B4674              <1>       mov    eax, [esi+LD_FreeSectors]
4249                              <1>
4250                              <1>       ;xor   ecx, ecx
4251                              <1>       ;dec   ecx ; 0FFFFFFFFh
4252                              <1>       ;cmp   eax, ecx ; 29/02/2016
4253                              <1>       ;je    short loc_get_free_sectors_retn ; recalculation is needed!
4254                              <1>
4255                              <1>       ; 23/03/2016
4256 0000CED9 8B4E70              <1>       mov    ecx, [esi+LD_TotalSectors]
4257 0000CEDC 39C1                <1>       cmp    ecx, eax ; Total sectors must be greater than Free sectors !
4258 0000CEDE 7707                <1>       ja     short loc_get_free_sectors_check_optype
4259                              <1>
4260 0000CEE0 31C0                <1>       xor    eax, eax
4261 0000CEE2 48                  <1>       dec    eax ; 0FFFFFFFFh  ; recalculation is needed!
4262 0000CEE3 894674              <1>       mov    [esi+LD_FreeSectors], eax ; reset (for recalculation)
4263                              <1>
4264                              <1> loc_get_free_sectors_retn:
4265 0000CEE6 C3                  <1>       retn
4266                              <1>
4267                              <1> loc_get_free_sectors_check_optype:
4268 0000CEE7 80FB03              <1>       cmp    bl, 3
4269 0000CEEA 7203                <1>       jb     short loc_set_fcc
4270                              <1>
4271 0000CEEC 29C9                <1>       sub    ecx, ecx ; 0
4272                              <1>
4273 0000CEEE C3                  <1>       retn
4274                              <1>
4275                              <1> loc_set_fcc:
4276 0000CEEF 807E0302            <1>       cmp    byte [esi+LD_FATType], 2
4277 0000CEF3 0F87DF000000        <1>       ja     loc_update_FAT32_fs_info_fcc
4278                              <1>
4279                              <1>       ;mov   eax, [esi+LD_FreeSectors]
4280 0000CEF9 0FB64E13            <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
4281 0000CEFD 29D2                <1>       sub    edx, edx
4282 0000CEFF F7F1                <1>       div    ecx
4283                              <1>       ;or    dx, dx
4284                              <1>       ;      ; DX -> Remain sectors < SecPerClust
4285                              <1>       ;      ; DX > 0 -> invalid free sector count
4286                              <1>       ;jnz   short loc_reset_fcc
4287                              <1>
4288                              <1> ;pass_set_fcc_div32:
4289 0000CF01 A3[9F880100]        <1>       mov    [FreeClusterCount], eax
4290 0000CF06 E988000000          <1>       jmp    loc_set_free_sectors_FAT12_FAT16
4291                              <1>
4292                              <1> loc_reset_fcc:
4293 0000CF0B 31C0                <1>       xor    eax, eax
4294 0000CF0D A3[9F880100]        <1>       mov    [FreeClusterCount], eax ; 0
4295 0000CF12 8B5678              <1>       mov    edx, [esi+LD_Clusters]
4296 0000CF15 42                  <1>       inc    edx
4297 0000CF16 8915[8E880100]      <1>       mov    [LastCluster], edx
4298                              <1>
4299 0000CF1C 807E0302            <1>       cmp    byte [esi+LD_FATType], 2
4300 0000CF20 7647                <1>       jna    short loc_count_free_fat_clusters_0
```

```
4301                                <1>
4302 0000CF22 48                    <1>       dec    eax ; FFFFFFFFh
4303 0000CF23 A3[308B0100]          <1>       mov    [CFS_FAT32FC], eax
4304                                <1>
4305                                <1>       ; 29/02/2016
4306 0000CF28 89463A                <1>       mov    [esi+LD_BPB+BPB_Reserved], eax ; reset
4307 0000CF2B 89463E                <1>       mov    [esi+LD_BPB+BPB_Reserved+4], eax ; reset
4308                                <1>
4309 0000CF2E B802000000            <1>       mov    eax, 2
4310                                <1>
4311                                <1> loc_count_fc_next_cluster_0:
4312 0000CF33 50                    <1>       push   eax
4313 0000CF34 E801F9FFFF            <1>       call   get_next_cluster
4314 0000CF39 7310                  <1>       jnc    short loc_check_fat32_ff_cluster
4315 0000CF3B 09C0                  <1>       or     eax, eax
4316 0000CF3D 741E                  <1>       jz     short pass_inc_cfs_fcc_0
4317                                <1>
4318                                <1> loc_put_fcc_unknown_sign:
4319 0000CF3F 58                    <1>       pop    eax
4320                                <1>       ; "Free count is Unknown" sign
4321                                <1>       ;mov   dword [FreeClusterCount], 0FFFFFFFFh
4322                                <1>
4323                                <1>       ; 29/02/2016
4324                                <1>       ; Save Free Cluster Count value in FAT32 'BPB_Reserved' area
4325                                <1>       ;mov   [esi+LD_BPB+BPB_Reserved], 0FFFFFFFFh ; unknown!
4326 0000CF40 8B15[308B0100]        <1>       mov    edx, [CFS_FAT32FC] ; First Free Cluster
4327                                <1>       ; Save First Free Cluster value in FAT32 'BPB_Reserved+4' area
4328 0000CF46 89563E                <1>       mov    [esi+LD_BPB+BPB_Reserved+4], edx
4329                                <1>
4330 0000CF49 EB7D                  <1>       jmp    loc_put_fcc_invalid_sign
4331                                <1>
4332                                <1> loc_check_fat32_ff_cluster:
4333 0000CF4B 09C0                  <1>       or     eax, eax
4334 0000CF4D 750E                  <1>       jnz    short pass_inc_cfs_fcc_0
4335 0000CF4F 58                    <1>       pop    eax
4336 0000CF50 A3[308B0100]          <1>       mov    [CFS_FAT32FC], eax
4337                                <1>       ;mov   dword [FreeClusterCount], 1
4338 0000CF55 FF05[9F880100]        <1>       inc    dword [FreeClusterCount]
4339 0000CF5B EB27                  <1>       jmp    short pass_inc_cfs_fcc_1
4340                                <1>
4341                                <1> pass_inc_cfs_fcc_0:
4342 0000CF5D 58                    <1>       pop    eax
4343                                <1>
4344                                <1> pass_inc_cfs_fcc_0c:
4345 0000CF5E 40                    <1>       inc    eax ; add eax, 1
4346 0000CF5F 3B05[8E880100]        <1>       cmp    eax, [LastCluster]
4347 0000CF65 76CC                  <1>       jna    short loc_count_fc_next_cluster_0
4348 0000CF67 EB6F                  <1>       jmp    short loc_update_FAT32_fs_info_fcc
4349                                <1>
4350                                <1> loc_count_free_fat_clusters_0:
4351                                <1>       ;mov   eax, 2
4352 0000CF69 B002                  <1>       mov    al, 2
4353                                <1>
4354                                <1> loc_count_fc_next_cluster:
4355 0000CF6B 50                    <1>       push   eax
4356 0000CF6C E8C9F8FFFF            <1>       call   get_next_cluster
4357 0000CF71 720C                  <1>       jc     short loc_count_fcc_stc
4358                                <1>
4359                                <1> loc_count_free_clusters_1:
4360 0000CF73 21C0                  <1>       and    eax, eax
4361 0000CF75 750C                  <1>       jnz    short pass_inc_cfs_fcc
4362                                <1>
4363 0000CF77 FF05[9F880100]        <1>       inc    dword [FreeClusterCount]
4364 0000CF7D EB04                  <1>       jmp    short pass_inc_cfs_fcc
4365                                <1>
4366                                <1> loc_count_fcc_stc:
4367 0000CF7F 09C0                  <1>       or     eax, eax
4368 0000CF81 75BC                  <1>       jnz    short loc_put_fcc_unknown_sign ; 29/02/2016
4369                                <1>
4370                                <1> pass_inc_cfs_fcc:
4371 0000CF83 58                    <1>       pop    eax
4372                                <1>
4373                                <1> pass_inc_cfs_fcc_1:
4374 0000CF84 40                    <1>       inc    eax ; add eax, 1
4375 0000CF85 3B05[8E880100]        <1>       cmp    eax, [LastCluster]
4376 0000CF8B 76DE                  <1>       jna    short loc_count_fc_next_cluster
4377                                <1>
4378                                <1> loc_set_free_sectors:
4379 0000CF8D 807E0302              <1>       cmp    byte [esi+LD_FATType], 2
4380 0000CF91 7745                  <1>       ja     short loc_update_FAT32_fs_info_fcc
4381                                <1>
4382                                <1> loc_set_free_sectors_FAT12_FAT16:
4383 0000CF93 803D[268B0100]00      <1>       cmp    byte [CFS_OPType], 0
4384 0000CF9A 761C                  <1>       jna    short pass_FAT_add_sub_fcc
4385 0000CF9C A1[288B0100]          <1>       mov    eax, [CFS_CC]
4386 0000CFA1 803D[268B0100]01      <1>       cmp    byte [CFS_OPType], 1
4387 0000CFA8 7708                  <1>       ja     short pass_FAT_add_fcc
4388 0000CFAA 0105[9F880100]        <1>       add    [FreeClusterCount], eax
4389 0000CFB0 EB06                  <1>       jmp    short pass_FAT_add_sub_fcc
4390                                <1>
4391                                <1> pass_FAT_add_fcc:
4392 0000CFB2 2905[9F880100]        <1>       sub    [FreeClusterCount], eax
4393                                <1>
4394                                <1> pass_FAT_add_sub_fcc:
4395 0000CFB8 0FB64613              <1>       movzx  eax, byte [esi+LD_BPB+SecPerClust]
4396 0000CFBC 8B15[9F880100]        <1>       mov    edx, [FreeClusterCount]
4397 0000CFC2 F7E2                  <1>       mul    edx
4398                                <1>
4399 0000CFC4 31C9                  <1>       xor    ecx, ecx
4400 0000CFC6 EB05                  <1>       jmp    short loc_cfs_retn_params
4401                                <1>
4402                                <1> loc_put_fcc_invalid_sign:
4403 0000CFC8 29C0                  <1>       sub    eax, eax ; 0
4404 0000CFCA 48                    <1>       dec    eax ; FFFFFFFFh
4405                                <1> loc_fat32_ffc_recalc_needed:
```

```
4406 0000CFCB 89C1                <1>       mov    ecx, eax
4407                              <1>
4408                              <1> loc_cfs_retn_params:
4409 0000CFCD 894674              <1>       mov    [esi+LD_FreeSectors], eax
4410 0000CFD0 0FB71D[268B0100]    <1>       movzx  ebx, word [CFS_OPType]
4411 0000CFD7 C3                  <1>       retn
4412                              <1>
4413                              <1> loc_update_FAT32_fs_info_fcc:
4414                              <1> loc_check_fcc_FSINFO_op:
4415                              <1>       ; 29/02/2016
4416                              <1>       ; EAX = Free cluster count (before this update) ; value from disk
4417                              <1>       ; EDX = First Free Cluster (before this update) ; value from disk
4418 0000CFD8 803D[268B0100]01    <1>       cmp    byte [CFS_OPType], 1
4419 0000CFDF 7221                <1>       jb     short loc_cfs_FAT32_get_rcalc_parms ; 0 = recalculated
4420 0000CFE1 7406                <1>       je     short loc_check_fcc_FSINFO_op1 ; 1 = add
4421                              <1> loc_check_fcc_FSINFO_op2: ; subtract
4422 0000CFE3 F71D[288B0100]      <1>       neg    dword [CFS_CC] ; prepare to subtract ; 2 = sub (add negative)
4423                              <1> loc_check_fcc_FSINFO_op1:
4424                              <1>       ; 01/03/2016
4425 0000CFE9 31D2                <1>       xor    edx, edx ; 0
4426 0000CFEB 4A                  <1>       dec    edx ; 0FFFFFFFFh
4427 0000CFEC 8B463A              <1>       mov    eax, [esi+LD_BPB+BPB_Reserved]
4428 0000CFEF 39D0                <1>       cmp    eax, edx
4429 0000CFF1 73D5                <1>       jnb    short loc_put_fcc_invalid_sign
4430 0000CFF3 0305[288B0100]      <1>       add    eax, [CFS_CC] ; free cluster count on disk + current count
4431 0000CFF9 72CD                <1>       jc     short loc_put_fcc_invalid_sign
4432                              <1>
4433 0000CFFB A3[9F880100]        <1>       mov    [FreeClusterCount], eax
4434 0000D000 EB0E                <1>       jmp    short loc_cfs_write_FSINFO_sector
4435                              <1>
4436                              <1> loc_cfs_FAT32_get_rcalc_parms:
4437 0000D002 8B15[308B0100]      <1>       mov    edx, [CFS_FAT32FC]
4438 0000D008 A1[9F880100]        <1>       mov    eax, [FreeClusterCount]
4439 0000D00D 89563E              <1>       mov    [esi+LD_BPB+BPB_Reserved+4], edx ; First Free Cluster
4440                              <1> loc_cfs_write_FSINFO_sector:
4441 0000D010 89463A              <1>       mov    [esi+LD_BPB+BPB_Reserved], eax ; Free cluster count
4442                              <1>       ; 01/03/2016
4443 0000D013 E8AA000000          <1>       call   set_fat32_fsinfo_sector_parms
4444 0000D018 72AE                <1>       jc     short loc_put_fcc_invalid_sign
4445                              <1>
4446                              <1> loc_set_FAT32_free_sectors:
4447                              <1>       ; 29/02/2016
4448                              <1>       ;mov   eax, [FreeClusterCount]
4449                              <1>       ;mov   ecx, eax
4450                              <1>       ;cmp   eax, 0FFFFFFFFh ; Invalid !
4451                              <1>       ;je    short loc_cfs_retn_params
4452                              <1>       ;
4453 0000D01A 8B0D[9F880100]      <1>       mov    ecx, [FreeClusterCount]
4454 0000D020 0FB64613            <1>       movzx  eax, byte [esi+LD_BPB+SecPerClust]
4455 0000D024 F7E1                <1>       mul    ecx
4456                              <1>       ; 29/02/2016
4457 0000D026 31C9                <1>       xor    ecx, ecx ; 0
4458 0000D028 09D2                <1>       or     edx, edx ; 0 ?
4459 0000D02A 759C                <1>       jnz    loc_put_fcc_invalid_sign
4460 0000D02C 394670              <1>       cmp    [esi+LD_TotalSectors], eax ; Volume size in sectors
4461 0000D02F 7697                <1>       jna    short loc_put_fcc_invalid_sign
4462                              <1>       ;
4463                              <1> loc_set_FAT32_free_sectors_ok:
4464 0000D031 31D2                <1>       xor    edx, edx ; 0
4465 0000D033 EB98                <1>       jmp    short loc_cfs_retn_params
4466                              <1>       ;
4467                              <1>
4468                              <1> get_last_cluster:
4469                              <1>       ; 22/10/2016
4470                              <1>       ; 27/02/2016 (TRDOS 386 =  TRDOS v2.0)
4471                              <1>       ; 12/06/2010 (DRV_FAT.ASM, 'proc_get_last_custer')
4472                              <1>       ; 06/06/2010
4473                              <1>       ; INPUT ->
4474                              <1>       ;      EAX = First Cluster Number
4475                              <1>       ;      ESI = Logical Dos Drive Parameters Table
4476                              <1>       ; OUTPUT ->
4477                              <1>       ;      cf = 0 -> No Error, EAX is valid
4478                              <1>       ;      cf = 1 -> EAX > 0 -> Error
4479                              <1>       ;      EAX = Last Cluster Number
4480                              <1>       ;       ECX = Previous Cluster -just before the last cluster-
4481                              <1>       ;       ; 22/10/2016
4482                              <1>       ;      [glc_index] = cluster index number of the last cluster
4483                              <1>       ;
4484                              <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
4485                              <1>
4486 0000D035 89C1                <1>       mov    ecx, eax
4487                              <1>
4488 0000D037 C705[388B0100]FFFF- <1>       mov    dword [glc_index], 0FFFFFFFFh ; 22/10/2016
4488 0000D03F FFFF                <1>
4489                              <1>
4490                              <1> loc_glc_get_next_cluster_1:
4491 0000D041 890D[348B0100]      <1>       mov    [glc_prevcluster], ecx
4492                              <1>       ; 22/10/2016
4493 0000D047 FF05[388B0100]      <1>       inc    dword [glc_index]
4494                              <1>
4495                              <1> loc_glc_get_next_cluster_2:
4496 0000D04D E8E8F7FFFF          <1>       call   get_next_cluster
4497                              <1>       ; ecx = current/previous cluster
4498                              <1>       ; eax = next/last cluster
4499 0000D052 73ED                <1>       jnc    short loc_glc_get_next_cluster_1
4500                              <1>
4501 0000D054 09C0                <1>       or     eax, eax
4502 0000D056 7509                <1>       jnz    short loc_glc_stc_retn
4503                              <1>
4504                              <1>       ; ecx = previous cluster
4505 0000D058 89C8                <1>         mov  eax, ecx
4506                              <1>
4507                              <1>       ; previous cluster becomes last cluster (ecx -> eax)
4508                              <1>       ; previous of previous cluster becomes previous cluster (ecx)
4509                              <1>
```

```
                                <1> loc_glc_prev_cluster_retn:
0000D05A 8B0D[348B0100]         <1>     mov     ecx, [glc_prevcluster]
0000D060 C3                     <1>     retn
                                <1>
                                <1> loc_glc_stc_retn:
0000D061 F5                     <1>     cmc     ;stc
0000D062 EBF6                   <1>     jmp     short loc_glc_prev_cluster_retn
                                <1>
                                <1> truncate_cluster_chain:
                                <1>     ; 01/03/2016
                                <1>     ; 28/02/2016 (TRDOS 386 =  TRDOS v2.0)
                                <1>     ; 22/01/2011 (DRV_FAT.ASM, 'proc_truncate_cluster_chain')
                                <1>     ; 11/09/2010
                                <1>     ; INPUT ->
                                <1>     ;     ESI = Logical dos drive description table address
                                <1>     ;     EAX = First cluster to be truncated/unlinked
                                <1>     ; OUTPUT ->
                                <1>     ;     ESI = Logical dos drive description table address
                                <1>     ;     ECX = Count of truncated/removed clusters
                                <1>     ;     CF = 0 -> EAX = Free sectors
                                <1>     ;     CF = 1 -> Error code in EAX (AL)
                                <1>
                                <1>     ; NOTE: This procedure does not update lm date&time !
                                <1>
                                <1> loc_truncate_cc:
0000D064 31C9                   <1>     xor     ecx, ecx ; mov ecx, 0
                                <1>     ;mov    byte [FAT_BuffValidData], 0
0000D066 890D[8A880100]         <1>     mov     [FAT_ClusterCounter], ecx ; 0 ; reset
                                <1>
                                <1> loc_tcc_unlink_clusters:
0000D06C E8F3FAFFFF             <1>     call    update_cluster
                                <1>     ; EAX = Next Cluster
                                <1>     ; ECX = Cluster Value
                                <1>     ; Note:
                                <1>     ; Returns count of unlinked clusters in
                                <1>     ; dword ptr FAT_ClusterCounter
0000D071 73F9                   <1>     jnc short loc_tcc_unlink_clusters
                                <1>
                                <1> pass_tcc_unlink_clusters:
0000D073 A2[3F8B0100]           <1>     mov     byte [TCC_FATErr], al
0000D078 803D[82880100]02       <1>     cmp     byte [FAT_BuffValidData], 2
0000D07F 750E                   <1>     jne     short loc_tcc_calculate_FAT_freespace
0000D081 E89BFDFFFF             <1>     call    save_fat_buffer
0000D086 7307                   <1>     jnc     short loc_tcc_calculate_FAT_freespace
0000D088 A2[3F8B0100]           <1>     mov     byte [TCC_FATErr], al ; Error
                                <1>     ;mov    byte [FAT_BuffValidData], 0
                                <1>
                                <1>     ; 01/03/2016
0000D08D EB12                   <1>     jmp     short loc_tcc_recalculate_FAT_freespace
                                <1>
                                <1> loc_tcc_calculate_FAT_freespace:
0000D08F A1[8A880100]           <1>     mov     eax, [FAT_ClusterCounter] ; signed (+-) number
0000D094 66BB01FF               <1>     mov     bx, 0FF01h ; BH = FFh -> ESI = Dos drv desc. table
                                <1>                        ; BL = 1 -> add cluster
0000D098 E819FEFFFF             <1>     call    calculate_fat_freespace
0000D09D 21C9                   <1>     and     ecx, ecx ; cx = 0 -> valid free sector count
0000D09F 7409                   <1>     jz      short pass_truncate_cc_recalc_FAT_freespace
                                <1>
                                <1> loc_tcc_recalculate_FAT_freespace:
0000D0A1 66BB00FF               <1>     mov     bx, 0FF00h ; recalculate !
0000D0A5 E80CFEFFFF             <1>     call    calculate_fat_freespace
                                <1>
                                <1> loc_tcc_calculate_FAT_freespace_err:
                                <1> pass_truncate_cc_recalc_FAT_freespace:
0000D0AA 8B0D[8A880100]         <1>     mov     ecx, [FAT_ClusterCounter]
                                <1>
0000D0B0 803D[3F8B0100]00       <1>     cmp     byte [TCC_FATErr], 0
0000D0B7 7608                   <1>     jna     short loc_tcc_unlink_clusters_retn
                                <1>
                                <1> loc_tcc_unlink_clusters_error:
0000D0B9 0FB605[3F8B0100]       <1>     movzx   eax, byte [TCC_FATErr]
0000D0C0 F9                     <1>     stc
                                <1> loc_tcc_unlink_clusters_retn:
0000D0C1 C3                     <1>     retn
                                <1>
                                <1> set_fat32_fsinfo_sector_parms:
                                <1>     ; 15/10/2016
                                <1>     ; 23/03/2016
                                <1>     ; 29/02/2016 (TRDOS 386 =  TRDOS v2.0)
                                <1>     ; INPUT ->
                                <1>     ;     ESI = Logical dos drive description table address
                                <1>     ;     [esi+LD_BPB+BPB_Reserved] = Free Cluster Count
                                <1>     ;     [esi+LD_BPB+BPB_Reserved+4] = First Free Cluster
                                <1>     ; OUTPUT ->
                                <1>     ;     ESI = Logical dos drive description table address
                                <1>     ;     CF = 0 -> OK..
                                <1>     ;     CF = 1 -> Error code in EAX (AL)
                                <1>     ;
                                <1>     ; (Modified registers: EAX, EBX, ECX, EDX)
                                <1>
0000D0C2 E824000000             <1>     call    get_fat32_fsinfo_sector_parms
0000D0C7 7221                   <1>     jc      short update_fat32_fsinfo_sector_retn
                                <1>
0000D0C9 8B463A                 <1>     mov     eax, [esi+LD_BPB+BPB_Reserved] ; Free Cluster Count
0000D0CC 8B563E                 <1>     mov     edx, [esi+LD_BPB+BPB_Reserved+4] ; First free Cluster
                                <1>
                                <1>     ;mov    ebx, DOSBootSectorBuff
0000D0CF 8983E8010000           <1>     mov     [ebx+488], eax
0000D0D5 8993EC010000           <1>     mov     [ebx+492], edx
                                <1>
0000D0DB A1[2C8B0100]           <1>     mov     eax, [CFS_FAT32FSINFOSEC]
0000D0E0 B901000000             <1>     mov     ecx, 1
0000D0E5 E8A2540000             <1>     call    disk_write
                                <1>     ;jnc    short update_fat32_fsinfo_sector_retn
                                <1>
```

```
                                <1>       ; 15/10/2016 (1Dh -> 18)
                                <1>       ; 23/03/2016 (1Dh)
                                <1>       ;mov   eax, 18 ; Drive not ready or write error
                                <1>
                                <1> update_fat32_fsinfo_sector_retn:
0000D0EA C3                     <1>       retn
                                <1>
                                <1> get_fat32_fsinfo_sector_parms:
                                <1>       ; 15/10/2016
                                <1>       ; 23/03/2016
                                <1>       ; 01/03/2016
                                <1>       ; 29/02/2016 (TRDOS 386 =  TRDOS v2.0)
                                <1>       ; INPUT ->
                                <1>       ;      ESI = Logical dos drive description table address
                                <1>       ; OUTPUT ->
                                <1>       ;      ESI = Logical dos drive description table address
                                <1>       ;      EBX = FSINFO sector buffer address (DOSBootSectorBuff)
                                <1>       ;      CF = 0 -> OK..
                                <1>       ;         EAX = FsInfo sector address
                                <1>       ;         ECX = Free cluster count
                                <1>       ;         EDX = First free cluster
                                <1>       ;      CF = 1 -> Error code in AL (EAX)
                                <1>       ;         EBX = 0
                                <1>       ;
                                <1>       ;      [CFS_FAT32FSINFOSEC] = FAT32 FSINFO sector address
                                <1>         ;
                                <1>       ; (Modified registers: EAX, EBX, ECX, EDX)
                                <1>
0000D0EB 0FB74636               <1>       movzx  eax, word [esi+LD_BPB+FAT32_FSInfoSec]
0000D0EF 03466C                 <1>       add    eax, [esi+LD_StartSector]
0000D0F2 A3[2C8B0100]           <1>       mov    [CFS_FAT32FSINFOSEC], eax
                                <1>
0000D0F7 BB[7E860100]           <1>       mov    ebx, DOSBootSectorBuff
0000D0FC B901000000             <1>       mov    ecx, 1
0000D101 E895540000             <1>       call   disk_read
0000D106 7232                   <1>       jc     short loc_read_FAT32_fsinfo_sec_err
                                <1>
0000D108 BB[7E860100]           <1>       mov    ebx, DOSBootSectorBuff
                                <1>
0000D10D 813B52526141           <1>       cmp    dword [ebx], 41615252h
0000D113 751E                   <1>       jne    short loc_read_FAT32_fsinfo_sec_stc
                                <1>
0000D115 81BBE4010000727241-    <1>       cmp    dword [ebx+484], 61417272h
0000D11E 61                     <1>
0000D11F 7512                   <1>       jne    short loc_read_FAT32_fsinfo_sec_stc
                                <1>
0000D121 A1[2C8B0100]           <1>       mov    eax, [CFS_FAT32FSINFOSEC]
0000D126 8B8BE8010000           <1>       mov    ecx, [ebx+488] ; free cluster count
0000D12C 8B93EC010000           <1>       mov    edx, [ebx+492] ; first (next) free cluster
                                <1>
0000D132 C3                     <1>       retn
                                <1>
                                <1> loc_read_FAT32_fsinfo_sec_stc:
                                <1>       ; 15/10/2016 (0Bh -> 28)
0000D133 B81C000000             <1>       mov    eax, 28 ; Invalid format!
0000D138 EB05                   <1>       jmp    short loc_read_FAT32_fsinfo_sec_stc_retn
                                <1>
                                <1> loc_read_FAT32_fsinfo_sec_err:
                                <1>       ; 15/10/2016 (15h -> 17)
                                <1>       ; 23/03/2016 (15h)
0000D13A B811000000             <1>       mov    eax, 17 ; Drive not ready or read error
                                <1>
                                <1> loc_read_FAT32_fsinfo_sec_stc_retn:
0000D13F 29DB                   <1>       sub    ebx, ebx ; 0
0000D141 F9                     <1>       stc
0000D142 C3                     <1>       retn
                                <1>
                                <1> add_new_cluster:
                                <1>       ; 15/10/2016
                                <1>       ; 16/05/2016
                                <1>       ; 18/03/2016, 24/03/2016
                                <1>       ; 11/03/2016 (TRDOS 386 =  TRDOS v2.0)
                                <1>       ; 30/07/2011 (DRV_FAT.ASM)
                                <1>       ; 11/09/2010
                                <1>       ; INPUT ->
                                <1>       ;      ESI = Logical dos drv desc. table address
                                <1>       ;      EAX = Last cluster
                                <1>       ; OUTPUT ->
                                <1>       ;      ESI = Logical dos drv desc. table address
                                <1>       ;      EAX = New Last cluster (next cluster)
                                <1>       ;      cf = 1 -> error code in EAX (AL)
                                <1>       ;      cf = 1 -> DX = sectors per cluster
                                <1>       ;      ECX = Free sectors
                                <1>       ; NOTE:
                                <1>       ; This procedure does not update lm date&time !
                                <1>       ;
                                <1>       ; (Modified registers: EAX, EBX, ECX, EDX, EDI)
                                <1>       ;
                                <1>
0000D143 A3[5C8C0100]           <1>       mov    [FAT_anc_LCluster], eax
                                <1>
0000D148 E844F9FFFF             <1>       call   get_first_free_cluster
0000D14D 720B                   <1>       jc     short loc_add_new_cluster_retn
                                <1>       ; EAX >= 2 and EAX < FFFFFFFFh is valid
                                <1>
0000D14F 89C2                   <1>       mov    edx, eax
                                <1>
0000D151 42                     <1>       inc    edx
                                <1>       ;jnz   short loc_add_new_cluster_check_ffc_eax
0000D152 7516                   <1>       jnz    short loc_add_new_cluster_save_fcc
                                <1>
                                <1> loc_add_new_cluster_no_disk_space_retn:
0000D154 B827000000             <1>       mov    eax, 27h ; MSDOS err => insufficient disk space
                                <1> loc_add_new_cluster_stc_retn:
0000D159 F9                     <1>       stc
```

```
                                <1> loc_add_new_cluster_retn:
0000D15A 0FB65E13               <1>       movzx  ebx, byte [esi+LD_BPB+SecPerClust]
0000D15E 8B4E74                 <1>       mov    ecx, [esi+LD_FreeSectors]
                                <1>       ;xor   edx, edx
                                <1>       ;stc
0000D161 C3                     <1>       retn
                                <1>
                                <1> loc_anc_invalid_format_stc_retn:
0000D162 F9                     <1>       stc
                                <1> loc_add_new_cluster_invalid_format_retn:
                                <1>       ; 15/10/2016 (0Bh -> 28)
0000D163 B81C000000             <1>       mov    eax, 28 ; Invalid format
0000D168 EBF0                   <1>       jmp    short loc_add_new_cluster_retn
                                <1>
                                <1> ;loc_add_new_cluster_check_ffc_eax:
                                <1> ;     cmp    eax, 2
                                <1> ;     jb     short loc_add_new_cluster_invalid_format_retn
                                <1>
                                <1> loc_add_new_cluster_save_fcc:
0000D16A A3[608C0100]           <1>       mov    [FAT_anc_FFCluster], eax
                                <1>
0000D16F 83E802                 <1>       sub    eax, 2
0000D172 0FB65E13               <1>       movzx  ebx, byte [esi+LD_BPB+SecPerClust]
0000D176 F7E3                   <1>       mul    ebx
0000D178 09D2                   <1>       or     edx, edx
0000D17A 75E6                   <1>       jnz    short loc_anc_invalid_format_stc_retn
                                <1>
                                <1> loc_add_new_cluster_allocate_cluster:
                                <1>       ; 18/03/2016
0000D17C 92                     <1>       xchg   edx, eax ; eax = 0
                                <1>       ; 16/05/2016
                                <1>       ;cmp   [ClusterBuffer_Valid], al ; 0
                                <1>       ;jna   short loc_anc_clear_cluster_buffer
                                <1>       ;; 'copy' command,
                                <1>       ;; writing destination file clust after reading source file clust
                                <1>       ;mov   [ClusterBuffer_Valid], al ; 0 ; reset
                                <1>       ;jmp   short loc_add_new_cluster_write_nc_to_disk
                                <1>
                                <1> loc_anc_clear_cluster_buffer:
                                <1>       ; 11/03/2016
                                <1>       ; Clear buffer
0000D17D BF00000700             <1>       mov    edi, Cluster_Buffer ; 70000h (for current TRDOS 386 version)
0000D182 89D9                   <1>       mov    ecx, ebx ; sector count
0000D184 C1E107                 <1>       shl    ecx, 7 ; 1 sector = 512 bytes -> 128 double words
                                <1>       ;xor   eax, eax ; 0
0000D187 F3AB                   <1>       rep    stosd
                                <1>
                                <1> loc_add_new_cluster_write_nc_to_disk:
                                <1>       ; 11/03/2016
                                <1>       ;xchg  eax, edx ; edx = 0, eax = sector offset
0000D189 89D0                   <1>       mov    eax, edx
0000D18B 034668                 <1>       add    eax, [esi+LD_DATABegin]
0000D18E 72D3                   <1>       jc     short loc_add_new_cluster_invalid_format_retn
                                <1>
0000D190 89D9                   <1>       mov    ecx, ebx ; ECX = sectors per cluster (<256)
0000D192 BB00000700             <1>       mov    ebx, Cluster_Buffer
0000D197 E8F0530000             <1>       call   disk_write
0000D19C 7307                   <1>       jnc    short loc_add_new_cluster_update_fat_nlc
                                <1>
                                <1>       ; 15/10/2016 (1Dh -> 18)
0000D19E B812000000             <1>       mov    eax, 18 ; Write Error
0000D1A3 EBB4                   <1>       jmp    short loc_add_new_cluster_stc_retn
                                <1>
                                <1> loc_add_new_cluster_update_fat_nlc:
0000D1A5 A1[608C0100]           <1>       mov    eax, [FAT_anc_FFCluster]
0000D1AA 31C9                   <1>       xor    ecx, ecx
0000D1AC 890D[8A880100]         <1>       mov    [FAT_ClusterCounter], ecx ; 0 ; reset
0000D1B2 49                     <1>       dec    ecx ; 0FFFFFFFFh
0000D1B3 E8ACF9FFFF             <1>       call   update_cluster
0000D1B8 7304                   <1>       jnc    short loc_add_new_cluster_update_fat_plc
0000D1BA 09C0                   <1>       or     eax, eax ;EAX = 0 -> cluster value is 0 or eocc
0000D1BC 759B                   <1>       jnz    short loc_add_new_cluster_stc_retn
                                <1>
                                <1> loc_add_new_cluster_update_fat_plc:
0000D1BE A1[5C8C0100]           <1>       mov    eax, [FAT_anc_LCluster]
0000D1C3 8B0D[608C0100]         <1>       mov    ecx, [FAT_anc_FFCluster]
0000D1C9 E896F9FFFF             <1>       call   update_cluster
0000D1CE 7314                   <1>       jnc    short loc_add_new_cluster_save_fat_buffer
0000D1D0 09C0                   <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
0000D1D2 7410                   <1>       jz     short loc_add_new_cluster_save_fat_buffer
                                <1>
                                <1> loc_anc_save_fat_buffer_err_retn:
                                <1>       ;cmp   byte [FAT_ClusterCounter], 1
                                <1>       ;jb    short loc_add_new_cluster_retn
                                <1>
0000D1D4 66BB00FF               <1>       mov    bx, 0FF00h ; recalculate free space (BL = 0)
                                <1>                          ; (BH = FFh -> Use ESI as Drv Param. Tbl.)
0000D1D8 50                     <1>       push   eax
0000D1D9 E8D8FCFFFF             <1>       call   calculate_fat_freespace
0000D1DE 58                     <1>       pop    eax
0000D1DF E975FFFFFF             <1>       jmp    loc_add_new_cluster_stc_retn
                                <1>
                                <1> loc_add_new_cluster_save_fat_buffer:
                                <1>       ;cmp   byte [FAT_BuffValidData], 2
                                <1>       ;jne   short loc_add_new_cluster_calc_FAT_freespace
                                <1>       ;Byte [FAT_BuffValidData] =  2
0000D1E4 E838FCFFFF             <1>       call   save_fat_buffer
0000D1E9 72E9                   <1>       jc     short loc_anc_save_fat_buffer_err_retn
                                <1>
                                <1> loc_add_new_cluster_calc_FAT_freespace:
                                <1>       ;mov   eax, 1 ; Only one Cluster
0000D1EB A1[8A880100]           <1>       mov    eax, [FAT_ClusterCounter]
0000D1F0 66BB01FF               <1>       mov    bx, 0FF01h ; BH = FFh -> ESI -> Dos drv desc. table
                                <1>               ; BL = 1 -> add cluster
0000D1F4 B301                   <1>       mov    bl, 01h ; BL = 1 -> add clusters
```

```
                                <1>       ; NOTE: EAX value will be added to Free Cluster Count
                                <1>       ; (Free Cluster Count is decreased when EAX value is negative)
0000D1F6 E8BBFCFFFF             <1>       call   calculate_fat_freespace
                                <1>       ;ECX = 0 -> no error, ECX > 0 -> error or invalid return
0000D1FB 21C9                   <1>       and    ecx, ecx ; ECX = 0 -> valid free sector count
0000D1FD 7409                   <1>       jz     short loc_add_new_cluster_return_cluster_number
                                <1>
                                <1> loc_add_new_cluster_recalc_FAT_freespace:
0000D1FF 66BB00FF               <1>       mov    bx, 0FF00h  ; recalculate free space
0000D203 E8AEFCFFFF             <1>       call   calculate_fat_freespace
                                <1>       ; cf = 0
                                <1> loc_add_new_cluster_return_cluster_number:
0000D208 89C1                   <1>       mov    ecx, eax ; Free sector count
0000D20A A1[608C0100]           <1>       mov    eax, [FAT_anc_FFCluster]
0000D20F 0FB65E13               <1>       movzx  ebx, byte [esi+LD_BPB+SecPerClust]
                                <1>       ;mov   edi, Cluster_Buffer
0000D213 31D2                   <1>       xor    edx, edx
0000D215 C3                     <1>       retn
                                <1>
                                <1> write_cluster:
                                <1>       ; 15/10/2016
                                <1>       ; 21/03/2016 (TRDOS 386 =  TRDOS v2.0)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;      EAX = Cluster Number (Sector index for SINGLIX FS)
                                <1>       ;      ESI = Logical DOS Drive Description Table address
                                <1>       ;      EBX = Cluster (File R/W) Buffer address (max. 64KB)
                                <1>       ;      Only for SINGLIX FS:
                                <1>       ;      EDX = File Number (The 1st FDT address)
                                <1>       ; OUTPUT ->
                                <1>       ;      cf = 1 -> Cluster can not be written onto disk
                                <1>       ;           EAX > 0 -> Error number
                                <1>       ;      cf = 0 -> Cluster has been written successfully
                                <1>       ;
                                <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
                                <1>
0000D216 0FB64E13               <1>       movzx  ecx, byte [esi+LD_BPB+BPB_SecPerClust]
                                <1>       ; CL = 1 = [esi+LD_FS_Reserved2] ; SectPerClust for Singlix FS
                                <1>
                                <1> write_file_sectors: ; 16/03/2016
0000D21A 807E0300               <1>       cmp    byte [esi+LD_FATType], 0
0000D21E 761C                   <1>       jna    short write_fs_cluster
                                <1>
                                <1> write_fat_file_sectors:
0000D220 83E802                 <1>       sub    eax, 2 ; Beginning cluster number is always 2
0000D223 0FB65613               <1>       movzx  edx, byte [esi+LD_BPB+BPB_SecPerClust] ; 18/03/2016
0000D227 F7E2                   <1>       mul    edx
0000D229 034668                 <1>       add    eax, [esi+LD_DATABegin] ; absolute address of the cluster
                                <1>
                                <1>       ; EAX = Disk sector address
                                <1>       ; ECX = Sector count
                                <1>       ; EBX = Buffer address
                                <1>       ; (EDX = 0)
                                <1>       ; ESI = Logical DOS drive description table address
                                <1>
0000D22C E85B530000             <1>       call   disk_write
0000D231 7306                   <1>       jnc    short wclust_retn
                                <1>
                                <1>       ; 15/10/2016 (1Dh -> 18)
0000D233 B812000000             <1>       mov    eax, 18 ; Drive not ready or write error !
0000D238 C3                     <1>       retn
                                <1>
                                <1> wclust_retn:
0000D239 29C0                   <1>       sub    eax, eax ; 0
0000D23B C3                     <1>       retn
                                <1>
                                <1> write_fs_cluster:
                                <1>       ; 21/03/2016 (TRDOS 386 =  TRDOS v2.0)
                                <1>       ; Singlix FS
                                <1>
                                <1>       ; EAX = Cluster number is sector index number of the file (eax)
                                <1>
                                <1>       ; EDX = File number is the first File Descriptor Table address
                                <1>       ;      of the file. (Absolute address of the FDT).
                                <1>
                                <1>       ; eax = sector index (0 for the first sector)
                                <1>       ; edx = FDT0 address
                                <1>             ; 64 KB buffer = 128 sectors (limit)
0000D23C B980000000             <1>       mov    ecx, 128 ; maximum count of sectors (before eof)
0000D241 E801000000             <1>       call   write_fs_sectors
0000D246 C3                     <1>       retn
                                <1>
                                <1> write_fs_sectors:
                                <1>       ; 21/03/2016 (TRDOS 386 =  TRDOS v2.0)
0000D247 F9                     <1>       stc
0000D248 C3                     <1>       retn
                                <1>
                                <1> get_cluster_by_index:
                                <1>       ; 29/04/2016 (TRDOS 386 =  TRDOS v2.0)
                                <1>       ; INPUT ->
                                <1>       ;      EAX = Beginning cluster
                                <1>       ;      EDX = Sector index in disk/file section
                                <1>       ;            (Only for SINGLIX file system!)
                                <1>       ;      ECX = Cluster sequence number after the beginning cluster
                                <1>       ;      ESI = Logical DOS Drive Description Table address
                                <1>       ; OUTPUT ->
                                <1>       ;      EAX = Cluster number
                                <1>       ;      cf = 1 -> Error code in AL (EAX)
                                <1>       ;
                                <1>       ;(Modified registers: EAX, ECX, EBX, EDX)
                                <1>       ;
0000D249 807E0301               <1>       cmp    byte [esi+LD_FATType], 1
0000D24D 721E                   <1>       jb     short get_fs_section_by_index
                                <1>
0000D24F 3B4E78                 <1>       cmp    ecx, [esi+LD_Clusters]
```

```
4929 0000D252 7207              <1>       jb     short gcbi_1
4930                            <1> gcbi_0:
4931 0000D254 F9                <1>       stc
4932 0000D255 B823000000        <1>       mov    eax, 23h ; Cluster not available !
4933                            <1>                       ; MSDOS error code: FCB unavailable
4934 0000D25A C3                <1>       retn
4935                            <1> gcbi_1:
4936 0000D25B 51                <1>       push   ecx
4937 0000D25C E8D9F5FFFF        <1>       call   get_next_cluster
4938 0000D261 59                <1>       pop    ecx
4939 0000D262 7203              <1>       jc     short gcbi_3
4940 0000D264 E2F5              <1>       loop   gcbi_1
4941                            <1> gcbi_2:
4942 0000D266 C3                <1>       retn
4943                            <1> gcbi_3:
4944 0000D267 09C0              <1>       or     eax, eax
4945 0000D269 74E9              <1>       jz     short gcbi_0
4946 0000D26B F5                <1>       cmc    ; stc
4947 0000D26C C3                <1>       retn
4948                            <1>
4949                            <1> get_fs_section_by_index:
4950                            <1>       ; 29/04/2016 (TRDOS 386 =  TRDOS v2.0)
4951                            <1>       ; INPUT ->
4952                            <1>       ;      EAX = Beginning FDT number/address
4953                            <1>       ;      EDX = Sector index in disk/file section
4954                            <1>       ;      ECX = Sector sequence number after the beginning FDT
4955                            <1>       ;      ESI = Logical DOS Drive Description Table address
4956                            <1>       ; OUTPUT ->
4957                            <1>       ;      EAX = FDT number/address
4958                            <1>       ;      EDX = Sector index of the section (0,1,2,3,4...)
4959                            <1>       ;      cf = 1 -> Error code in AL (EAX)
4960                            <1>       ;
4961                            <1>       ;(Modified registers: EAX, ECX, EBX, EDX)
4962                            <1>       ;
4963 0000D26D B8FFFFFFFF        <1>       mov    eax, 0FFFFFFFFh
4964 0000D272 C3                <1>       retn
4965                            <1>
4966                            <1> get_last_section:
4967                            <1>       ; 22/10/2016 (TRDOS 386 =  TRDOS v2.0)
4968                            <1>       ; INPUT ->
4969                            <1>       ;      EAX = (The 1st) FDT number/address
4970                            <1>       ;      ESI = Logical DOS Drive Description Table address
4971                            <1>       ; OUTPUT ->
4972                            <1>       ;      EAX = FDT number/address of the last section
4973                            <1>       ;      EDX = Last sector of the section (0,1,2,3,4...)
4974                            <1>       ;      [glc_index] = sector index number of the last sector
4975                            <1>       ;                    (for file, not for the last section)
4976                            <1>       ;
4977                            <1>       ;      cf = 1 -> Error code in AL (EAX)
4978                            <1>       ;
4979                            <1>       ;(Modified registers: EAX, ECX, EBX, EDX)
4980                            <1>       ;
4981 0000D273 B800000000        <1>       mov    eax, 0
4982 0000D278 BA00000000        <1>       mov    edx, 0
4983 0000D27D C3                <1>       retn
3111                                %include 'trdosk6.s' ; 24/01/2016
3112                            <1> ; ****************************************************************************
3113                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.4) - MAIN PROGRAM : trdosk6.s
3114                            <1> ; ----------------------------------------------------------------------------
3115                            <1> ; Last Update: 17/04/2021
3116                            <1> ; ----------------------------------------------------------------------------
3117                            <1> ; Beginning: 24/01/2016
3118                            <1> ; ----------------------------------------------------------------------------
3119                            <1> ; Assembler: NASM version 2.15 t(trdos386.s)
3120                            <1> ; ----------------------------------------------------------------------------
3121                            <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
3122                            <1> ; u1.s (27/17/2015), u2.s (03/01/2016)
3123                            <1> ; ****************************************************************************
3124                            <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
3125                            <1> ; TRDOS2.ASM (09/11/2011)
3126                            <1> ; ----------------------------------------------------------------------------
3127                            <1> ; INT_21H.ASM (c) 2009-2011 Erdogan TAN  [14/11/2009] Last Update: 08/11/2011
3128                            <1>
3129                            <1> sysent: ; < enter to system call >
3130                            <1>       ; 17/03/2017
3131                            <1>       ; 03/03/2017
3132                            <1>       ; 19/02/2017
3133                            <1>       ; 13/01/2017
3134                            <1>       ; 06/06/2016
3135                            <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
3136                            <1>       ; 16/04/2015 - 19/10/2015 (Retro UNIX 386 v1)
3137                            <1>       ; 10/04/2013 - 18/01/2014 (Retro UNIX 8086 v1)
3138                            <1>       ;
3139                            <1>       ; 'unkni' or 'sysent' is sytem entry from various traps.
3140                            <1>       ; The trap type is determined and an indirect jump is made to
3141                            <1>       ; the appropriate system call handler. If there is a trap inside
3142                            <1>       ; the system a jump to panic is made. All user registers are saved
3143                            <1>       ; and u.sp points to the end of the users stack. The sys (trap)
3144                            <1>       ; instructor is decoded to get the the system code part (see
3145                            <1>       ; trap instruction in the PDP-11 handbook) and from this
3146                            <1>       ; the indirect jump address is calculated. If a bad system call is
3147                            <1>       ; made, i.e., the limits of the jump table are exceeded, 'badsys'
3148                            <1>       ; is called. If the call is legitimate control passes to the
3149                            <1>       ; appropriate system routine.
3150                            <1>       ;
3151                            <1>       ; Calling sequence:
3152                            <1>       ;     Through a trap caused by any sys call outside the system.
3153                            <1>       ; Arguments:
3154                            <1>       ;     Arguments of particular system call.
3155                            <1>       ; ......................................................
3156                            <1>       ;
3157                            <1>       ; Retro UNIX 8086 v1 modification:
3158                            <1>       ;       System call number is in EAX register.
3159                            <1>       ;
3160                            <1>       ;       Other parameters are in EDX, EBX, ECX, ESI, EDI, EBP
```

```
3161                                    <1>     ;       registers depending of function details.
3162                                    <1>     ;
3163                                    <1>     ; 16/04/2015
3164 0000D27E 368925[5C030300]          <1>     mov     [ss:u.sp], esp ; Kernel stack points to return address
3165                                    <1>
3166                                    <1>     ; save user registers
3167 0000D285 1E                        <1>     push    ds
3168 0000D286 06                        <1>     push    es
3169 0000D287 0FA0                      <1>     push    fs
3170 0000D289 0FA8                      <1>     push    gs
3171 0000D28B 60                        <1>     pushad  ; eax, ecx, edx, ebx, esp -before pushad-, ebp, esi, edi
3172                                    <1>     ;
3173                                    <1>     ; ESPACE = [ss:u.sp] - esp ; 4*12 = 48 ; 17/09/2015 ; 06/06/2016
3174                                    <1>     ;       (ESPACE is size of space in kernel stack
3175                                    <1>     ;       for saving/restoring user registers.)
3176                                    <1>     ;
3177 0000D28C 50                        <1>     push    eax ; 01/07/2015
3178 0000D28D 66B81000                  <1>     mov     ax, KDATA
3179 0000D291 8ED8                      <1>     mov     ds, ax
3180 0000D293 8EC0                      <1>     mov     es, ax
3181 0000D295 8EE0                      <1>     mov     fs, ax
3182 0000D297 8EE8                      <1>     mov     gs, ax
3183 0000D299 A1[A8800100]              <1>     mov     eax, [k_page_dir]
3184 0000D29E 0F22D8                    <1>     mov     cr3, eax
3185 0000D2A1 58                        <1>     pop     eax ; 01/07/2015
3186                                    <1>     ; 19/10/2015
3187 0000D2A2 FC                        <1>     cld
3188                                    <1>     ;
3189 0000D2A3 FE05[5B030300]            <1>     inc     byte [sysflg]
3190                                    <1>             ; incb sysflg / indicate a system routine is in progress
3191 0000D2A9 FB                        <1>     sti     ; 18/01/2014
3192 0000D2AA 0F85FB9DFFFF              <1>     jnz      panic ; 24/05/2013
3193                                    <1>             ; beq 1f
3194                                    <1>             ; jmp panic ; / called if trap inside system
3195                                    <1> ;1:
3196                                    <1>     ; 17/03/2017
3197 0000D2B0 80642438FE                <1>     and     byte [esp+ESPACE+8], ~1 ; clear carry flag
3198                                    <1>
3199                                    <1>     ; 16/04/2015
3200 0000D2B5 A3[64030300]              <1>     mov     [u.r0], eax
3201 0000D2BA 8925[60030300]            <1>     mov     [u.usp], esp ; kernel stack points to user's registers
3202                                    <1>
3203                                    <1>     ; 13/01/2017 (TRDOS 386 Feaure only !)
3204 0000D2C0 803D[D4030300]00          <1>     cmp     byte [u.t_lock], 0 ; timer interrupt lock ?
3205 0000D2C7 0F879D010000              <1>     ja      sysrele                 ; yes, sys release only !!!
3206                                    <1>
3207                                    <1>             ; mov $s.syst+2,clockp
3208                                    <1>             ; mov r0,-(sp) / save user registers
3209                                    <1>             ; mov sp,u.r0 / pointer to bottom of users stack
3210                                    <1>                     ; / in u.r0
3211                                    <1>             ; mov r1,-(sp)
3212                                    <1>             ; mov r2,-(sp)
3213                                    <1>             ; mov r3,-(sp)
3214                                    <1>             ; mov r4,-(sp)
3215                                    <1>             ; mov r5,-(sp)
3216                                    <1>             ; mov ac,-(sp) / "accumulator" register for extended
3217                                    <1>                     ; / arithmetic unit
3218                                    <1>             ; mov mq,-(sp) / "multiplier quotient" register for the
3219                                    <1>                     ; / extended arithmetic unit
3220                                    <1>             ; mov sc,-(sp) / "step count" register for the extended
3221                                    <1>                     ; / arithmetic unit
3222                                    <1>             ; mov sp,u.sp / u.sp points to top of users stack
3223                                    <1>             ; mov 18.(sp),r0 / store pc in r0
3224                                    <1>             ; mov -(r0),r0 / sys inst in r0      10400xxx
3225                                    <1>             ; sub $sys,r0 / get xxx code
3226 0000D2CD C1E002                    <1>     shl     eax, 2
3227                                    <1>             ; asl r0 / multiply by 2 to jump indirect in bytes
3228 0000D2D0 3DB8000000              <1>     cmp     eax, end_of_syscalls - syscalls
3229                                    <1>             ; cmp r0,$2f-1f / limit of table (35) exceeded
3230                                    <1>     ;jnb    short badsys
3231                                    <1>             ; bhis badsys / yes, bad system call
3232 0000D2D5 F5                        <1>     cmc
3233 0000D2D6 9C                        <1>     pushf
3234 0000D2D7 50                        <1>     push    eax
3235 0000D2D8 8B2D[5C030300]            <1>     mov     ebp, [u.sp] ; Kernel stack at the beginning of sys call
3236 0000D2DE B0FE                      <1>     mov     al, 0FEh ; 11111110b
3237 0000D2E0 1400                      <1>     adc     al, 0 ; al = al + cf
3238 0000D2E2 204508                    <1>     and     [ebp+8], al ; flags (reset carry flag)
3239                                    <1>             ; bic $341,20.(sp) / set users processor priority to 0
3240                                    <1>                             ; / and clear carry bit
3241 0000D2E5 5D                        <1>     pop     ebp ; eax
3242 0000D2E6 9D                        <1>     popf
3243 0000D2E7 0F8208020000              <1>     jc      badsys
3244 0000D2ED A1[64030300]              <1>     mov     eax, [u.r0]
3245                                    <1>     ; system call registers: EAX, EDX, ECX, EBX, ESI, EDI
3246 0000D2F2 FFA5[F8D20000]            <1>     jmp     dword [ebp+syscalls]
3247                                    <1>             ; jmp *1f(r0) / jump indirect thru table of addresses
3248                                    <1>                           ; / to proper system routine.
3249                                    <1> syscalls: ; 1:
3250                                    <1>     ; 31/12/2017
3251                                    <1>     ; 28/02/2017
3252                                    <1>     ; 20/02/2017
3253                                    <1>     ; 19/02/2017
3254                                    <1>     ; 15/10/2016
3255                                    <1>     ; 20/05/2016
3256                                    <1>     ; 19/05/2016
3257                                    <1>     ; 16/05/2016
3258                                    <1>     ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
3259                                    <1>     ; 21/09/2015
3260                                    <1>     ; 01/07/2015
3261                                    <1>     ; 16/04/2015 (32 bit address modification)
3262 0000D2F8 [86140100]                <1>     dd sysver    ; 0 ; Get TRDOS 386 version number (v2.0)
3263 0000D2FC [57D50000]                <1>     dd sysexit   ; 1
3264 0000D300 [2CD70000]                <1>     dd sysfork   ; 2
3265 0000D304 [CADA0000]                <1>     dd sysread   ; 3
```

```
3266 0000D308 [E9DA0000]          <1>       dd syswrite  ; 4
3267 0000D30C [15D90000]          <1>       dd sysopen   ; 5
3268 0000D310 [A1DA0000]          <1>       dd sysclose  ; 6
3269 0000D314 [AED60000]          <1>       dd syswait   ; 7
3270 0000D318 [44D80000]          <1>       dd syscreat  ; 8
3271 0000D31C [DA220100]          <1>       dd sysrename ; 9  ; TRDOS 386, Rename File (31/12/2017)
3272 0000D320 [551E0100]          <1>       dd sysdelete ; 10 ; TRDOS 386, Delete File (29/12/2017)
3273 0000D324 [3A080100]          <1>       dd sysexec   ; 11
3274 0000D328 [7F1F0100]          <1>       dd syschdir  ; 12
3275 0000D32C [44210100]          <1>       dd systime   ; 13 ; TRDOS 386, Get Sys Date&Time (30/12/2017)
3276 0000D330 [63DA0000]          <1>       dd sysmkdir  ; 14
3277 0000D334 [B31F0100]          <1>       dd syschmod  ; 15 ; TRDOS 386, Change Attributes (30/12/2017)
3278 0000D338 [BC1E0100]          <1>       dd sysrmdir  ; 16 ; TRDOS 386, Remove Directory (29/12/2017)
3279 0000D33C [FE0A0100]          <1>       dd sysbreak  ; 17
3280 0000D340 [99200100]          <1>       dd sysdrive  ; 18 ; TRDOS 386, Get/Set Current Drv (30/12/2017)
3281 0000D344 [3F0B0100]          <1>       dd sysseek   ; 19
3282 0000D348 [510B0100]          <1>       dd systell   ; 20
3283 0000D34C [FB230100]          <1>       dd sysmem    ; 21 ; TRDOS 386, Get Total&Free Mem (31/12/2017)
3284 0000D350 [31240100]          <1>       dd sysprompt ; 22 ; TRDOS 386, Change Cmd Prompt (31/12/2017)
3285 0000D354 [73240100]          <1>       dd syspath   ; 23 ; TRDOS 386, Get/Set Run Path (31/12/2017)
3286 0000D358 [E0240100]          <1>       dd sysenv    ; 24 ; TRDOS 386, Get/Set Env Vars (31/12/2017)
3287 0000D35C [C5210100]          <1>       dd sysstime  ; 25 ; TRDOS 386, Set Sys Date&Time (30/12/2017)
3288 0000D360 [B70B0100]          <1>       dd sysquit   ; 26
3289 0000D364 [AB0B0100]          <1>       dd sysintr   ; 27
3290 0000D368 [E8200100]          <1>       dd sysdir    ; 28 ; TRDOS 386, Get Curr Drive&Dir (30/12/2017)
3291 0000D36C [46DB0000]          <1>       dd sysemt    ; 29
3292 0000D370 [23210100]          <1>       dd sysldrvt  ; 30 ; TRDOS 386, Get Logical DOS DDT (30/12/2017)
3293 0000D374 [F7DC0000]          <1>       dd sysvideo  ; 31 ; TRDOS 386 Video Functions (16/05/2016)
3294 0000D378 [5E2D0100]          <1>       dd sysaudio  ; 32 ; TRDOS 386 Audio Functions (16/05/2016)
3295 0000D37C [5FDB0000]          <1>       dd systimer  ; 33 ; TRDOS 386 Timer Functions (18/05/2016)
3296 0000D380 [F80B0100]          <1>       dd syssleep  ; 34 ; Retro UNIX 8086 v1 feature only !
3297                              <1>                            ; 11/06/2014
3298 0000D384 [270C0100]          <1>       dd sysmsg    ; 35 ; Retro UNIX 386 v1 feature only !
3299                              <1>                            ; 01/07/2015
3300 0000D388 [440D0100]          <1>       dd sysgeterr ; 36 ; Retro UNIX 386 v1 feature only !
3301                              <1>                            ; 21/09/2015 - get last error number
3302 0000D38C [2C1E0100]          <1>       dd sysfpstat ; 37 ; TRDOS 386 FPU state option (28/02/2017)
3303 0000D390 [95140100]          <1>       dd syspri    ; 38 ; change priority - TRDOS 386 (20/05/2016)
3304 0000D394 [6AD40000]          <1>       dd sysrele   ; 39 ; TRDOS 386 (19/05/2016) (0 -> 39)
3305 0000D398 [C8150100]          <1>       dd sysfff    ; 40 ; Find First File - TRDOS 386 (15/10/2016)
3306 0000D39C [A7160100]          <1>       dd sysfnf    ; 41 ; Find Next File - TRDOS 386 (15/10/2016)
3307 0000D3A0 [171D0100]          <1>       dd sysalloc  ; 42 ; Allocate contiguous memory block/pages
3308                              <1>                            ; TRDOS 386 (19/02/2017) DMA buff fuctions
3309 0000D3A4 [D51D0100]          <1>       dd sysdalloc ; 43 ; Deallocate contiguous memory block/pages
3310                              <1>                            ; TRDOS 386 (19/02/2017) DMA buff fuctions
3311 0000D3A8 [101E0100]          <1>       dd syscalbac ; 44 ; IRQ Callback and Signal Response Byte
3312                              <1>                            ; service setup - TRDOS 386 (20/02/2017)
3313                              <1>                            ; 28/08/2017 (20/08/2017)
3314 0000D3AC [EF350100]          <1>       dd sysdma    ; 45 ; TRDOS 386 - (ISA) DMA service
3315                              <1>
3316                              <1> end_of_syscalls:
3317                              <1>
3318                              <1> error:
3319                              <1>       ; 18/05/2016
3320                              <1>       ; 13/05/2016
3321                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
3322                              <1>       ; 16/04/2015 - 17/09/2015 (Retro UNIX 386 v1)
3323                              <1>       ; 10/04/2013 - 07/08/2013 (Retro UNIX 8086 v1)
3324                              <1>       ;
3325                              <1>       ; 'error' merely sets the error bit off the processor status (c-bit)
3326                              <1>       ; then falls right into the 'sysret', 'sysrele' return sequence.
3327                              <1>       ;
3328                              <1>       ; INPUTS -> none
3329                              <1>       ; OUTPUTS ->
3330                              <1>       ;     processor status - carry (c) bit is set (means error)
3331                              <1>       ;
3332                              <1>       ; 26/05/2013 (Stack pointer must be reset here!
3333                              <1>       ;           Because, jumps to error procedure
3334                              <1>       ;           disrupts push-pop nesting balance)
3335                              <1>       ;
3336 0000D3B0 8B2D[5C030300]      <1>       mov    ebp, [u.sp] ; interrupt (system call) return (iretd) address
3337 0000D3B6 804D0801            <1>       or     byte [ebp+8], 1  ; set carry bit of flags register
3338                              <1>                               ; (system call will return with cf = 1)
3339                              <1>             ; bis $1,20.(r1) / set c bit in processor status word below
3340                              <1>                               ; / users stack
3341                              <1>       ; 17/09/2015
3342 0000D3BA 83ED30              <1>       sub    ebp, ESPACE ; 48 ; total size of stack frame ('sysdefs.inc')
3343                              <1>                               ; for saving/restoring user registers
3344                              <1>       ;cmp   ebp, [u.usp]
3345                              <1>       ;je    short err0
3346 0000D3BD 892D[60030300]      <1>       mov    [u.usp], ebp
3347                              <1> ;err0:
3348                              <1>       ; 01/09/2015
3349 0000D3C3 8B25[60030300]      <1>       mov    esp, [u.usp]     ; Retro Unix 8086 v1 modification!
3350                              <1>                               ; 10/04/2013
3351                              <1>                               ; (If an I/O error occurs during disk I/O,
3352                              <1>                               ; related procedures will jump to 'error'
3353                              <1>                               ; procedure directly without returning to
3354                              <1>                               ; the caller procedure. So, stack pointer
3355                              <1>                                     ; must be restored here.)
3356                              <1>       ; 13/05/2016
3357                              <1>       ; NOTE: (The last) error code is in 'u.error', it can be retrieved by
3358                              <1>       ;     'get last error' system call later.
3359                              <1>
3360                              <1>       ; 03/09/2015 - 09/06/2015 - 07/08/2013
3361 0000D3C9 C605[C6030300]00    <1>       mov    byte [u.kcall], 0 ; namei_r, mkdir_w reset
3362                              <1>
3363                              <1> sysret: ; < return from system call>
3364                              <1>       ; 01/03/2017
3365                              <1>       ; 28/02/2017
3366                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
3367                              <1>       ; 16/04/2015 - 10/09/2015 (Retro UNIX 386 v1)
3368                              <1>       ; 10/04/2013 - 23/02/2014 (Retro UNIX 8086 v1)
3369                              <1>       ;
3370                              <1>       ; 'sysret' first checks to see if process is about to be
```

```
                                <1>       ; terminated (u.bsys). If it is, 'sysexit' is called.
                                <1>       ; If not, following happens:
                                <1>       ;     1) The user's stack pointer is restored.
                                <1>       ;     2) r1=0 and 'iget' is called to see if last mentioned
                                <1>       ;        i-node has been modified. If it has, it is written out
                                <1>       ;        via 'ppoke'.
                                <1>       ;     3) If the super block has been modified, it is written out
                                <1>       ;        via 'ppoke'.
                                <1>       ;     4) If the dismountable file system's super block has been
                                <1>       ;        modified, it is written out to the specified device
                                <1>       ;        via 'ppoke'.
                                <1>       ;     5) A check is made if user's time quantum (uquant) ran out
                                <1>       ;        during his execution. If so, 'tswap' is called to give
                                <1>       ;        another user a chance to run.
                                <1>       ;     6) 'sysret' now goes into 'sysrele'.
                                <1>       ;         (See 'sysrele' for conclusion.)
                                <1>       ;
                                <1>       ; Calling sequence:
                                <1>       ;     jump table or 'br sysret'
                                <1>       ; Arguments:
                                <1>       ;     -
                                <1>       ; ............................................................
                                <1>       ;
                                <1>       ; ((AX=r1 for 'iget' input))
                                <1>       ;
0000D3D0 31C0                   <1>       xor   eax, eax ; 28/02/2017
                                <1> sysret0: ; 29/07/2015 (eax = 0, jump from sysexec)
0000D3D2 FEC0                   <1>       inc   al ; 04/05/2013
0000D3D4 3805[B2030300]         <1>       cmp   [u.bsys], al ; 1
                                <1>             ; tstb u.bsys / is a process about to be terminated because
0000D3DA 0F8377010000           <1>       jnb   sysexit ; 04/05/2013
                                <1>             ; bne sysexit / of an error? yes, go to sysexit
                                <1>       ;mov   esp, [u.usp] ; 24/05/2013 (that is not needed here)
                                <1>             ; mov u.sp,sp / no point stack to users stack
0000D3E0 FEC8                   <1>       dec   al ; mov ax, 0
                                <1>             ; clr r1 / zero r1 to check last mentioned i-node
0000D3E2 E8A4510000             <1>       call  iget
                                <1>             ; jsr r0,iget / if last mentioned i-node has been modified
                                <1>                           ; / it is written out
                                <1>       ; 10/01/2017
                                <1>       ; 09/01/2017
                                <1> ;sysrele: ; < release >
                                <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; 16/04/2015 - 14/10/2015 (Retro UNIX 386 v1)
                                <1>       ; 10/04/2013 - 07/03/2014 (Retro UNIX 8086 v1)
                                <1>       ;
                                <1>       ; 'sysrele' first calls 'tswap' if the time quantum for a user is
                                <1>       ;  zero (see 'sysret'). It then restores the user's registers and
                                <1>       ; turns off the system flag. It then checked to see if there is
                                <1>       ; an interrupt from the user by calling 'isintr'. If there is,
                                <1>       ; the output gets flashed (see isintr) and interrupt action is
                                <1>       ; taken by a branch to 'intract'. If there is no interrupt from
                                <1>       ; the user, a rti is made.
                                <1>       ;
                                <1>       ; Calling sequence:
                                <1>       ;     Fall through a 'bne' in 'sysret' & ?
                                <1>       ; Arguments:
                                <1>       ;     -
                                <1>       ; ............................................................
                                <1>       ;
                                <1>       ; 23/02/2014 (swapret)
                                <1>       ; 22/09/2013
                                <1> sysrel0: ;1:
0000D3E7 803D[A8030300]00       <1>       cmp   byte [u.quant], 0 ; 16/05/2013
                                <1>             ; tstb uquant / is the time quantum 0?
0000D3EE 7705                   <1>       ja    short swapret
                                <1>             ; bne 1f / no, don't swap it out
                                <1> sysrelease: ; 07/12/2013 (jump from 'clock')
0000D3F0 E8593F0000             <1>       call  tswap
                                <1>             ; jsr r0,tswap / yes, swap it out
                                <1>
                                <1> ; Retro Unix 8086 v1 feature: return from 'swap' to 'swapret' address.
                                <1> swapret: ;1:
                                <1>       ; 10/09/2015
                                <1>       ; 01/09/2015
                                <1>       ; 14/05/2015
                                <1>       ; 16/04/2015 (Retro UNIX 386 v1 - 32 bit, pm modifications)
                                <1>       ; 26/05/2013 (Retro UNIX 8086 v1)
                                <1>       ; cli
                                <1>       ; 24/07/2015
                                <1>       ;
                                <1>       ;; 'esp' must be already equal to '[u.usp]' here !
                                <1>       ;; mov esp, [u.usp]
                                <1>
                                <1>       ; 22/09/2013
0000D3F5 E891510000             <1>       call  isintr
                                <1>       ; 20/10/2013
0000D3FA 7405                   <1>       jz    short sysrel1
0000D3FC E83F010000             <1>       call  intract
                                <1>             ; jsr r0,isintr / is there an interrupt from the user
                                <1>             ;     br intract / yes, output gets flushed, take interrupt
                                <1>                           ; / action
                                <1> sysrel1:
0000D401 FA                     <1>       cli   ; 14/10/2015
                                <1> sysrel2:
                                <1>       ; 28/02/2017
                                <1>       ; Check if there is a (delayed) callback for current user/process
0000D402 A0[D7030300]           <1>       mov   al, [u.irqwait]
0000D407 240F                   <1>       and   al, 0Fh ; is there a waiting IRQ callback service ?
0000D409 7444                   <1>       jz    short sysrel8 ; no
                                <1>
                                <1>       ; Set return to IRQ callback service and return from the service
0000D40B 0FB6D8                 <1>       movzx ebx, al
0000D40E 883D[D7030300]         <1>       mov   [u.irqwait], bh ; 0 ; reset
0000D414 8A9B[70400100]         <1>       mov   bl, [ebx+IRQenum] ; (available) IRQ index +1 (1 to 9)
```

```
3476                                    <1>       ; 01/03/2017
3477 0000D41A FECB                      <1>       dec    bl ; IRQ index number, 0 to 8
3478 0000D41C 7831                      <1>       js     short sysrel8 ; 0 -> FFh (not in use!?)
3479                                    <1>       ;
3480 0000D41E A0[B3030300]              <1>       mov    al, [u.uno] ; current process (user) number
3481 0000D423 3883[888F0100]            <1>       cmp    [ebx+IRQ.owner], al
3482 0000D429 7524                      <1>       jne    short sysrel8 ; it is not the current user/process !?
3483 0000D42B F683[9A8F0100]01          <1>       test   byte [ebx+IRQ.method], 1 ; callback ?
3484 0000D432 741B                      <1>       jz     short sysrel8 ; not a callback method !?
3485                                    <1>
3486 0000D434 8B93[AC8F0100]            <1>       mov    edx, [ebx+IRQ.addr] ; IRQ callback service address (virtual)
3487 0000D43A C605[D8030300]01          <1>       mov    byte [u.r_lock], 1 ; IRQ callback service in progress flag
3488                                    <1>
3489 0000D441 E8B03F0000                <1>       call   wswap ; save user's registers & status
3490                                    <1>              ;       (for return from IRQ callback service)
3491                                    <1>
3492 0000D446 8B2D[5C030300]            <1>       mov    ebp, [u.sp]; kernel's stack, points to EIP (user)
3493 0000D44C 895500                    <1>       mov    [ebp], edx ; IRQ call back service address
3494                                    <1> sysrel8:
3495 0000D44F FE0D[5B030300]            <1>       dec    byte [sysflg]
3496                                    <1>              ; decb sysflg / turn system flag off
3497                                    <1>
3498 0000D455 A1[B8030300]              <1>       mov    eax, [u.pgdir]
3499 0000D45A 0F22D8                    <1>       mov    cr3, eax  ; 1st PDE points to Kernel Page Table 0 (1st 4 MB)
3500                                    <1>                      ; (others are different than kernel page tables)
3501                                    <1>       ; 10/09/2015
3502 0000D45D 61                        <1>       popad ; edi, esi, ebp, temp (icrement esp by 4), ebx, edx, ecx, eax
3503                                    <1>             ; mov (sp)+,sc / restore user registers
3504                                    <1>             ; mov (sp)+,mq
3505                                    <1>             ; mov (sp)+,ac
3506                                    <1>             ; mov (sp)+,r5
3507                                    <1>             ; mov (sp)+,r4
3508                                    <1>             ; mov (sp)+,r3
3509                                    <1>             ; mov (sp)+,r2
3510                                    <1>       ;
3511 0000D45E A1[64030300]              <1>       mov    eax, [u.r0]  ; ((return value in EAX))
3512 0000D463 0FA9                      <1>       pop    gs
3513 0000D465 0FA1                      <1>       pop    fs
3514 0000D467 07                        <1>       pop    es
3515 0000D468 1F                        <1>       pop    ds
3516                                    <1>       ;or    word [esp+8], 200h ; 22/01/2017 ; force enabling interrupts
3517 0000D469 CF                        <1>       iretd
3518                                    <1>             ; rti / no, return from interrupt
3519                                    <1>
3520                                    <1> sysrele:
3521                                    <1>       ; 24/03/2017
3522                                    <1>       ; 28/02/2017
3523                                    <1>       ; 27/02/2017
3524                                    <1>       ; 29/01/2017
3525                                    <1>       ; 14/01/2017
3526                                    <1>       ; 13/01/2017
3527                                    <1>       ; 09/01/2017, 10/01/2017, 12/01/2017
3528                                    <1>       ; Major modification for TRDOS 386 (CallBack return)
3529                                    <1>       ;
3530                                    <1>       ; 'sysrele' system call restores previously saved
3531                                    <1>       ; registers and addresses of the process
3532                                    <1>       ; (Main purpose -in TRDOS 386- is to return from
3533                                    <1>       ; timer callback service routine in ring 3 -user mode-.)
3534                                    <1>       ;
3535                                    <1>       ; check if the process is in timer callback phase
3536 0000D46A 803D[D4030300]00          <1>       cmp    byte [u.t_lock], 0 ; TIMER INT LOCK
3537                                    <1>       ;je    short sysrel0 ; classic (Retro UNIX 386 type) sysrele
3538 0000D471 7734                      <1>       ja     short sysrel3
3539                                    <1>       ; 27/02/2017
3540 0000D473 803D[D8030300]00          <1>       cmp    byte [u.r_lock], 0 ; IRQ callback lock
3541 0000D47A 0F8667FFFFFF              <1>       jna    sysrel0 ; classic sysrele ; 24/03/2017
3542 0000D480 E859000000                <1>       call   sysrel7
3543 0000D485 803D[D8030300]00          <1>       cmp    byte [u.r_lock], 0 ; IRQ callback service lock
3544 0000D48C 7628                      <1>       jna    short sysrel4
3545 0000D48E C605[D8030300]00          <1>       mov    byte [u.r_lock], 0 ; reset
3546                                    <1>       ;mov   byte [u.irqwait], 0 ; reset ; 28/02/2017
3547 0000D495 A0[D9030300]              <1>       mov    al, [u.r_mode]
3548 0000D49A 08C0                      <1>       or     al, al
3549 0000D49C 7518                      <1>       jnz    short sysrel4
3550 0000D49E FEC8                      <1>       dec    al
3551 0000D4A0 A2[D9030300]              <1>       mov    [u.r_mode], al ; 0FFh ; not necessary !?
3552 0000D4A5 EB32                      <1>       jmp    short sysrel6
3553                                    <1> sysrel3:
3554                                    <1>       ; 27/02/2017
3555 0000D4A7 E832000000                <1>       call   sysrel7
3556                                    <1>       ; 14/01/2017
3557 0000D4AC 28C0                      <1>       sub    al, al
3558 0000D4AE 3805[D4030300]            <1>       cmp    [u.t_lock], al ; 0 ; TIMER INT LOCK
3559 0000D4B4 770E                      <1>       ja     short sysrel5 ; yes
3560                                    <1> sysrel4:
3561                                    <1>       ; 29/01/2017
3562 0000D4B6 8B44241C                  <1>       mov    eax, [esp+28] ; eax
3563 0000D4BA A3[64030300]              <1>       mov    [u.r0], eax
3564 0000D4BF E93EFFFFFF                <1>       jmp    sysrel2
3565                                    <1> sysrel5:
3566 0000D4C4 A2[D4030300]              <1>       mov    [u.t_lock], al ; 0 ; reset
3567 0000D4C9 A0[D5030300]              <1>       mov    al, [u.t_mode]
3568 0000D4CE 20C0                      <1>       and    al, al
3569                                    <1>       ;jnz   short sysrel2 ; 0FFh ; user mode
3570 0000D4D0 75E4                      <1>       jnz    short sysrel4 ; 29/01/2017
3571 0000D4D2 FEC8                      <1>       dec    al
3572 0000D4D4 A2[D5030300]              <1>       mov    [u.t_mode], al ; 0FFh ; not necessary !?
3573                                    <1> sysrel6:
3574                                    <1>       ; cpu will continue from the interrupted sytem call addr
3575 0000D4D9 61                        <1>       popad        ; edi, esi, ebp, esp, ebx, edx, ecx, eax
3576 0000D4DA 83C410                    <1>       add    esp, 16      ; pass segment segisters: ds, es, fs, gs
3577 0000D4DD CF                        <1>       iretd        ; eip, cs, eflags
3578                                    <1>
3579                                    <1> sysrel7:
3580 0000D4DE 0FB61D[B3030300]          <1>       movzx  ebx, byte [u.uno] ; current process number
```

```
3581 0000D4E5 66C1E302            <1>       shl    bx, 2
3582                              <1>       ;cmp   [ebx+p.tcb-4], eax ; 0 ; is there callback address ?
3583                              <1>       ;jna   short sysrel0
3584                              <1>       ; yes, reset callback address then restore process registers
3585                              <1>       ;mov   [ebx+p.tcb-4], eax ; 0 ; reset
3586 0000D4E9 8B83[BC000300]      <1>       mov    eax, [ebx+p.upage-4] ; UPAGE address
3587 0000D4EF FA                  <1>       cli    ; disable interrupts till 'iretd'
3588 0000D4F0 E9393F0000          <1>       jmp    rswap ; restore process 'u' structure
3589                              <1>
3590                              <1> badsys:
3591                              <1>       ; 25/12/2016
3592                              <1>       ; 18/04/2016 (TRDOS 386 = TRDOS v2.0)
3593                              <1>       ; 17/04/2011 (TRDOS v1.0, 'IFC.ASM')
3594                              <1>       ; 03/02/2011 ('trdos_ifc_routine')
3595                              <1>       ;
3596                              <1>       ; 16/04/2015 (Retro UNIX 386 v1, 'badsys')
3597                              <1>       ; (EIP, EAX values will be shown on screen with error message)
3598                              <1>       ; (EIP = 'CD 40h' instruction address -INT 40h-)
3599                              <1>       ; (EAX = Function number)
3600                              <1>       ;
3601 0000D4F5 FE05[B2030300]      <1>       inc    byte [u.bsys]
3602                              <1>       ;
3603 0000D4FB 8B1D[5C030300]      <1>       mov    ebx, [u.sp] ; esp at the beginning of 'sysent'
3604 0000D501 8B03                <1>       mov    eax, [ebx] ; EIP (return address, not 'INT 30h' address)
3605 0000D503 83E802              <1>       sub    eax, 2 ; CDh, ##h
3606 0000D506 E8446DFFFF          <1>       call   dwordtohex
3607 0000D50B 8915[63400100]      <1>       mov    [eip_str], edx
3608 0000D511 A3[67400100]        <1>       mov    [eip_str+4], eax
3609 0000D516 A1[64030300]        <1>       mov    eax, [u.r0]
3610 0000D51B E82F6DFFFF          <1>       call   dwordtohex
3611 0000D520 8915[52400100]      <1>       mov    [eax_str], edx
3612 0000D526 A3[56400100]        <1>       mov    [eax_str+4], eax
3613                              <1>
3614 0000D52B 66C705[47400100]34- <1>       mov    word [int_num_str], SYSCALL_INT_NUM ; 25/12/2016
3614 0000D533 30                  <1>
3615                              <1>
3616 0000D534 BE[19400100]        <1>       mov    esi, ifc_msg ; "invalid funtion call !" msg (trdosk9.s)
3617 0000D539 E8079BFFFF          <1>       call   print_msg
3618                              <1>
3619 0000D53E EB17                <1>       jmp    sysexit
3620                              <1>
3621                              <1> intract: ; / interrupt action
3622                              <1>       ; 14/10/2015
3623                              <1>       ; 16/04/2015 (Retro UNIX 386 v1 - Beginning)
3624                              <1>       ; 09/05/2013 - 07/12/2013 (Retro UNIX 8086 v1)
3625                              <1>       ;
3626                              <1>       ; Retro UNIX 8086 v1 modification !
3627                              <1>       ; (Process/task switching and quit routine by using
3628                              <1>       ; Retro UNIX 8086 v1 keyboard interrupt output.))
3629                              <1>       ;
3630                              <1>       ; input -> 'u.quit'  (also value of 'u.intr' > 0)
3631                              <1>       ; output -> If value of 'u.quit' = FFFFh ('ctrl+brk' sign)
3632                              <1>       ;            'intract' will jump to 'sysexit'.
3633                              <1>       ;          Intract will return to the caller
3634                              <1>       ;            if value of 'u.quit' <> FFFFh.
3635                              <1>       ; 14/10/2015
3636 0000D540 FB                  <1>       sti
3637                              <1>       ; 07/12/2013
3638 0000D541 66FF05[AC030300]    <1>       inc    word [u.quit]
3639 0000D548 7408                <1>       jz     short intrct0 ; FFFFh -> 0
3640 0000D54A 66FF0D[AC030300]    <1>       dec    word [u.quit]
3641                              <1>       ; 16/04/2015
3642 0000D551 C3                  <1>       retn
3643                              <1> intrct0:
3644 0000D552 58                  <1>       pop    eax ; call intract -> retn
3645                              <1>       ;
3646 0000D553 31C0                <1>       xor    eax, eax
3647 0000D555 FEC0                <1>       inc    al  ; mov ax, 1
3648                              <1> ;;;
3649                              <1>       ; UNIX v1 original 'intract' routine...
3650                              <1>       ; / interrupt action
3651                              <1>             ;cmp *(sp),$rti / are you in a clock interrupt?
3652                              <1>             ; bne 1f / no, 1f
3653                              <1>             ; cmp (sp)+,(sp)+ / pop clock pointer
3654                              <1>       ; 1: / now in user area
3655                              <1>             ; mov r1,-(sp) / save r1
3656                              <1>             ; mov u.ttyp,r1
3657                              <1>                   ; / pointer to tty buffer in control-to r1
3658                              <1>             ; cmpb 6(r1),$177
3659                              <1>                   ; / is the interrupt char equal to "del"
3660                              <1>             ; beq 1f / yes, 1f
3661                              <1>             ; clrb 6(r1)
3662                              <1>                    ; / no, clear the byte
3663                              <1>                   ; / (must be a quit character)
3664                              <1>             ; mov (sp)+,r1 / restore r1
3665                              <1>             ; clr u.quit / clear quit flag
3666                              <1>             ; bis $20,2(sp)
3667                              <1>                   ; / set trace for quit (sets t bit of
3668                              <1>                   ; / ps-trace trap)
3669                              <1>             ; rti   ;  / return from interrupt
3670                              <1>       ; 1: / interrupt char = del
3671                              <1>             ; clrb 6(r1) / clear the interrupt byte
3672                              <1>                     ; / in the buffer
3673                              <1>             ; mov (sp)+,r1 / restore r1
3674                              <1>             ; cmp u.intr,$core / should control be
3675                              <1>                           ; / transferred to loc core?
3676                              <1>             ; blo 1f
3677                              <1>             ; jmp *u.intr / user to do rti yes,
3678                              <1>                           ; / transfer to loc core
3679                              <1>       ; 1:
3680                              <1>             ; sys 1 / exit
3681                              <1>
3682                              <1> sysexit: ; <terminate process>
3683                              <1>       ; 14/11/2017
3684                              <1>       ; 27/05/2017
```

```
                                <1>       ; 10/04/2017
                                <1>       ; 26/02/2017, 28/02/2017
                                <1>       ; 02/01/2017, 23/01/2017
                                <1>       ; 06/06/2016, 10/06/2016
                                <1>       ; 19/05/2016, 23/05/2016
                                <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; 16/04/2015 - 01/09/2015 (Retro UNIX 386 v1)
                                <1>       ; 19/04/2013 - 14/02/2014 (Retro UNIX 8086 v1)
                                <1>       ;
                                <1>       ; 'sysexit' terminates a process. First each file that
                                <1>       ; the process has opened is closed by 'flose'. The process
                                <1>       ; status is then set to unused. The 'p.pid' table is then
                                <1>       ; searched to find children of the dying process. If any of
                                <1>       ; children are zombies (died by not waited for), they are
                                <1>       ; set free. The 'p.pid' table is then searched to find the
                                <1>       ; dying process's parent. When the parent is found, it is
                                <1>       ; checked to see if it is free or it is a zombie. If it is
                                <1>       ; one of these, the dying process just dies. If it is waiting
                                <1>       ; for a child process to die, it notified that it doesn't
                                <1>       ; have to wait anymore by setting it's status from 2 to 1
                                <1>       ; (waiting to active). It is awakened and put on runq by
                                <1>       ; 'putlu'. The dying process enters a zombie state in which
                                <1>       ; it will never be run again but stays around until a 'wait'
                                <1>       ; is completed by it's parent process. If the parent is not
                                <1>       ; found, process just dies. This means 'swap' is called with
                                <1>       ; 'u.uno=0'. What this does is the 'wswap' is not called
                                <1>       ; to write out the process and 'rswap' reads the new process
                                <1>       ; over the one that dies..i.e., the dying process is
                                <1>       ; overwritten and destroyed.
                                <1>       ;
                                <1>       ; Calling sequence:
                                <1>       ;     sysexit or conditional branch.
                                <1>       ; Arguments:
                                <1>       ;     -
                                <1>       ; ..........................................................
                                <1>       ;
                                <1>       ; Retro UNIX 8086 v1 modification:
                                <1>       ;       System call number (=1) is in EAX register.
                                <1>       ;
                                <1>       ;       Other parameters are in EDX, EBX, ECX, ESI, EDI, EBP
                                <1>       ;       registers depending of function details.
                                <1>       ;
                                <1>       ; ('swap' procedure is mostly different than original UNIX v1.)
                                <1>       ;
                                <1> ; / terminate process
                                <1>       ; AX = 1
0000D557 6648                   <1>       dec    ax ; 0
0000D559 66A3[AA030300]         <1>       mov    [u.intr], ax ; 0
                                <1>              ; clr u.intr / clear interrupt control word
                                <1>              ; clr r1 / clear r1
                                <1> sysexit_0:
                                <1>       ; 23/01/2017
                                <1>       ; 02/01/2017
                                <1>       ; 10/06/2016
                                <1>       ; 06/06/2016
                                <1>       ; 23/05/2016
                                <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; Check and stop/clear timer event(s) of this (dying) process
                                <1>       ; if there is.
                                <1>
                                <1>       ; 02/01/2017
0000D55F FA                     <1>       cli    ; disable interrupts
                                <1>       ; 23/01/2017 - reset timer frequency (to 18.2Hz)
0000D560 B036                   <1>       mov    al, 00110110b ; 36h
0000D562 E643                   <1>       out    43h, al
0000D564 28C0                   <1>       sub    al, al ; 0
0000D566 E640                   <1>       out    40h, al ; LB
0000D568 E640                   <1>       out    40h, al ; HB
                                <1>       ;
0000D56A 0FB61D[B3030300]       <1>       movzx  ebx, byte [u.uno]
                                <1>       ;mov   bl, [u.uno] ; process number of dying process
0000D571 3883[FF000300]         <1>       cmp    byte [ebx+p.timer-1], al ; 0
0000D577 763A                   <1>       jna    short sysexit_12 ; no timer events for this process
0000D579 8883[FF000300]         <1>       mov    byte [ebx+p.timer-1], al ; 0 ; reset
                                <1>       ;mov   al, [timer_events]
                                <1>       ;or    al, al
                                <1>       ;jz    short sysexit_12 ; no timer events
                                <1>       ;mov   cl, al
0000D57F 8A0D[3B8D0100]         <1>       mov    cl, [timer_events] ; 14/11/2017
                                <1>       ;cli   ; disable interrupts
0000D585 B410                   <1>       mov    ah, 16 ; number of available timer events
0000D587 BE[60040300]           <1>       mov    esi, timer_set ; beginning address of timer events
                                <1> sysexit_7:
0000D58C 8A06                   <1>       mov    al, [esi] ; process number (of timer event)
0000D58E 38D8                   <1>       cmp    al, bl ; process number comparison
0000D590 7411                   <1>       je     short sysexit_10
0000D592 20C0                   <1>       and    al, al
0000D594 7404                   <1>       jz     short sysexit_9
                                <1> sysexit_8:
0000D596 FEC9                   <1>       dec    cl
0000D598 7416                   <1>       jz     short sysexit_11
                                <1> sysexit_9:
0000D59A FECC                   <1>       dec    ah
0000D59C 7415                   <1>       jz     short sysexit_12
0000D59E 83C610                 <1>       add    esi, 16
0000D5A1 EBE9                   <1>       jmp    short sysexit_7
                                <1>
                                <1> sysexit_10:
                                <1>       ;mov   byte [esi], 0
0000D5A3 66C7060000             <1>       mov    word [esi], 0
                                <1>       ;mov   dword [esi+12], 0
                                <1>       ;
0000D5A8 FE0D[3B8D0100]         <1>       dec    byte [timer_events] ; 02/01/2017
                                <1>       ;
0000D5AE EBE6                   <1>       jmp    short sysexit_8
```

```
3790                                    <1>
3791                                    <1> sysexit_11:
3792 0000D5B0 6629C0                    <1>         sub    ax, ax ; 0 ; 26/02/2017
3793                                    <1> sysexit_12:
3794                                    <1>         ; 26/02/2017 (Unlink IRQ callbacks belong to the user)
3795 0000D5B3 803D[D6030300]00          <1>         cmp    byte [u.irqc], 0 ; Count of IRQ callbacks
3796 0000D5BA 7E2E                      <1>         jng    short sysexit_16 ; zero or invalid
3797                                    <1>         ; 28/02/2017
3798                                    <1>         ; clear IRQ callback flags (for 'sysrele' and 'sysret')
3799 0000D5BC A2[D7030300]              <1>         mov    [u.irqwait], al ; 0 ; force to clear waiting flag
3800 0000D5C1 A2[D8030300]              <1>         mov    [u.r_lock], al ; 0 ; force to clear busy flag
3801 0000D5C6 BE[888F0100]              <1>         mov    esi, IRQ.owner
3802                                    <1> sysexit_13:
3803 0000D5CB AC                        <1>         lodsb
3804 0000D5CC 3A05[B3030300]            <1>         cmp    al, [u.uno] ; owner = current user ?
3805 0000D5D2 750C                      <1>         jne    short sysexit_14
3806 0000D5D4 C646FF00                  <1>         mov    byte [esi-1], 0 ; owner = 0 : Free
3807 0000D5D8 FE0D[D6030300]            <1>         dec    byte [u.irqc]
3808 0000D5DE 7408                      <1>         jz     short sysexit_15
3809                                    <1> sysexit_14:
3810 0000D5E0 81FE[908F0100]            <1>         cmp    esi, IRQ.owner + 8 ; the last IRQ index number ?
3811 0000D5E6 76E3                      <1>         jna    short sysexit_13 ; no
3812                                    <1> sysexit_15:
3813 0000D5E8 30C0                      <1>         xor    al, al ; 0
3814                                    <1> sysexit_16: ; 2:
3815 0000D5EA FB                        <1>         sti    ; enable interrupts
3816                                    <1>         ;
3817                                    <1>         ; AX = 0
3818                                    <1> sysexit_1: ; 1:
3819                                    <1>         ; AX = File descriptor
3820                                    <1>               ; / r1 has file descriptor (index to u.fp list)
3821                                    <1>               ; / Search the whole list
3822 0000D5EB E842340000                <1>         call   fclose
3823                                    <1>               ; jsr r0,fclose / close all files the process opened
3824                                    <1>         ;; ignore error return
3825                                    <1>               ; br .+2 / ignore error return
3826                                    <1>         ;inc   ax
3827 0000D5F0 FEC0                      <1>         inc    al
3828                                    <1>               ; inc r1 / increment file descriptor
3829                                    <1>         ;cmp   ax, 10
3830 0000D5F2 3C0A                      <1>         cmp    al, 10
3831                                    <1>               ; cmp r1,$10. / end of u.fp list?
3832 0000D5F4 72F5                      <1>         jb     short sysexit_1
3833                                    <1>               ; blt 1b / no, go back
3834                                    <1>         ;movzx ebx, byte [u.uno]
3835 0000D5F6 8A1D[B3030300]            <1>         mov    bl, [u.uno] ; 02/01/2017
3836                                    <1>               ; movb u.uno,r1 / yes, move dying process's number to r1
3837 0000D5FC 88A3[AF000300]            <1>         mov    [ebx+p.stat-1], ah ; 0, SFREE
3838                                    <1>               ; clrb p.stat-1(r1) / free the process
3839                                    <1>         ; 10/04/2017
3840 0000D602 381D[01900100]            <1>         cmp    [audio_user], bl
3841 0000D608 7518                      <1>         jne    short sysexit_17
3842                                    <1>         ; reset audio device (current) owner and 'initializated' flag
3843 0000D60A 883D[01900100]            <1>         mov    [audio_user], bh ; 0
3844                                    <1>         ; 27/05/2017
3845 0000D610 8B0D[EC8F0100]            <1>         mov    ecx,  [audio_buffer]
3846 0000D616 09C9                      <1>         or     ecx, ecx
3847 0000D618 7408                      <1>         jz     short sysexit_17
3848                                    <1>         ; 'deallocate_user_pages' is not necessary in sysexit !!!
3849                                    <1>         ;push  ebx
3850                                    <1>         ;mov   ebx, ecx
3851                                    <1>         ;mov   ecx, [audio_buff_size]
3852                                    <1>         ;call  deallocate_user_pages
3853                                    <1>         ;; (Modified Registers -> EAX, EDX, ESI, EDI, EBX, ECX, EBP)
3854 0000D61A 29C9                      <1>         sub    ecx, ecx
3855 0000D61C 890D[EC8F0100]            <1>         mov    [audio_buffer], ecx ; 0
3856                                    <1>         ;pop   ebx
3857                                    <1> sysexit_17:
3858                                    <1>         ;shl   bx, 1
3859 0000D622 D0E3                      <1>         shl    bl, 1
3860                                    <1>               ; asl r1 / use r1 for index into the below tables
3861 0000D624 668B8B[1E000300]          <1>         mov    cx, [ebx+p.pid-2]
3862                                    <1>               ; mov p.pid-2(r1),r3 / move dying process's name to r3
3863 0000D62B 668B93[3E000300]          <1>         mov    dx, [ebx+p.ppid-2]
3864                                    <1>               ; mov p.ppid-2(r1),r4 / move its parents name to r4
3865                                    <1>         ; xor  bx, bx ; 0
3866 0000D632 30DB                      <1>         xor    bl, bl ; 0
3867                                    <1>               ; clr r2
3868 0000D634 31F6                      <1>         xor    esi, esi ; 0
3869                                    <1>               ; clr r5 / initialize reg
3870                                    <1> sysexit_2: ; 1:
3871                                    <1>                ; / find children of this dying process,
3872                                    <1>               ; / if they are zombies, free them
3873                                    <1>         ;add   bx, 2
3874 0000D636 80C302                    <1>         add    bl, 2
3875                                    <1>               ; add $2,r2 / search parent process table
3876                                    <1>                           ; / for dying process's name
3877 0000D639 66398B[3E000300]          <1>         cmp    [ebx+p.ppid-2], cx
3878                                    <1>               ; cmp p.ppid-2(r2),r3 / found it?
3879 0000D640 7513                      <1>         jne    short sysexit_4
3880                                    <1>               ; bne 3f / no
3881                                    <1>         ;shr   bx, 1
3882 0000D642 D0EB                      <1>         shr    bl, 1
3883                                    <1>               ; asr r2 / yes, it is a parent
3884 0000D644 80BB[AF000300]03          <1>         cmp    byte [ebx+p.stat-1], 3 ; SZOMB
3885                                    <1>               ; cmpb p.stat-1(r2),$3 / is the child of this
3886                                    <1>                                    ; / dying process a zombie
3887 0000D64B 7506                      <1>         jne    short sysexit_3
3888                                    <1>               ; bne 2f / no
3889 0000D64D 88A3[AF000300]            <1>         mov    [ebx+p.stat-1], ah ; 0, SFREE
3890                                    <1>               ; clrb p.stat-1(r2) / yes, free the child process
3891                                    <1> sysexit_3: ; 2:
3892                                    <1>         ;shr   bx, 1
3893 0000D653 D0E3                      <1>         shl    bl, 1
3894                                    <1>               ; asl r2
```

```
                                <1> sysexit_4: ; 3:
                                <1>             ; / search the process name table
                                <1>             ; / for the dying process's parent
0000D655 663993[1E000300]       <1>       cmp   [ebx+p.pid-2], dx
                                <1>             ; cmp p.pid-2(r2),r4 / found it?
0000D65C 7502                   <1>       jne   short sysexit_5
                                <1>             ; bne 3f / no
0000D65E 89DE                   <1>       mov   esi, ebx
                                <1>             ; mov r2,r5 / yes, put index to p.pid table (parents
                                <1>                       ; / process # x2) in r5
                                <1> sysexit_5: ; 3:
                                <1>       ;cmp  bx, nproc + nproc
0000D660 80FB20                 <1>       cmp   bl, nproc + nproc
                                <1>             ; cmp r2,$nproc+nproc / has whole table been searched?
0000D663 72D1                   <1>       jb    short sysexit_2
                                <1>             ; blt 1b / no, go back
                                <1>             ; mov r5,r1 / yes, r1 now has parents process # x2
0000D665 21F6                   <1>       and   esi, esi ; r5=r1
0000D667 7436                   <1>       jz    short sysexit_6
                                <1>             ; beq 2f / no parent has been found.
                                <1>                    ; / The process just dies
0000D669 66D1EE                 <1>       shr   si, 1
                                <1>             ; asr r1 / set up index to p.stat
0000D66C 8A86[AF000300]         <1>       mov   al, [esi+p.stat-1]
                                <1>             ; movb p.stat-1(r1),r2 / move status of parent to r2
0000D672 20C0                   <1>       and   al, al
0000D674 7429                   <1>       jz    short sysexit_6
                                <1>             ; beq 2f / if its been freed, 2f
0000D676 3C03                   <1>       cmp   al, 3
                                <1>             ; cmp r2,$3 / is parent a zombie?
0000D678 7425                   <1>       je    short sysexit_6
                                <1>             ; beq 2f / yes, 2f
                                <1>       ; BH = 0
0000D67A 8A1D[B3030300]         <1>       mov   bl, [u.uno]
                                <1>             ; movb u.uno,r3 / move dying process's number to r3
0000D680 C683[AF000300]03       <1>       mov   byte [ebx+p.stat-1], 3  ; SZOMB
                                <1>             ; movb $3,p.stat-1(r3) / make the process a zombie
0000D687 3C01                   <1>       cmp   al, 1 ; SRUN
0000D689 7414                   <1>       je    short sysexit_6
                                <1>       ;cmp  al, 2
                                <1>             ; cmp r2,$2 / is the parent waiting for
                                <1>                    ; / this child to die
                                <1>       ;jne  short sysexit_6
                                <1>             ; bne 2f / yes, notify parent not to wait any more
                                <1>       ; p.stat = 2 --> waiting
                                <1>       ; p.stat = 4 --> sleeping
0000D68B C686[AF000300]01       <1>       mov   byte [esi+p.stat-1], 1 ; SRUN
                                <1>       ;dec  byte [esi+p.stat-1]
                                <1>             ; decb p.stat-1(r1) / awaken it by putting it (parent)
0000D692 6689F0                 <1>       mov   ax, si ; r1  (process number in AL)
                                <1>       ;
                                <1>       ;mov  ebx, runq + 4
                                <1>             ; mov $runq+4,r2 / on the runq
0000D695 BB[54030300]           <1>       mov   ebx, runq+2 ; normal run queue ; 02/01/2017
0000D69A E8C73D0000             <1>       call  putlu
                                <1>             ; jsr r0, putlu
                                <1> sysexit_6:
                                <1>             ; / the process dies
0000D69F C605[B3030300]00       <1>       mov   byte [u.uno], 0
                                <1>             ; clrb u.uno / put zero as the process number,
                                <1>                 ; / so "swap" will
0000D6A6 E8BD3C0000             <1>       call  swap
                                <1>             ; jsr r0,swap / overwrite process with another process
                                <1> hlt_sys:
                                <1>       ;sti
                                <1> hlts0:
0000D6AB F4                     <1>       hlt
0000D6AC EBFD                   <1>       jmp   short hlts0
                                <1>             ; 0 / and thereby kill it; halt?
                                <1>
                                <1> syswait: ; < wait for a processs to die >
                                <1>       ; 17/09/2015
                                <1>       ; 02/09/2015
                                <1>       ; 01/09/2015
                                <1>       ; 16/04/2015 (Retro UNIX 386 v1 - Beginning)
                                <1>       ; 24/05/2013 - 05/02/2014 (Retro UNIX 8086 v1)
                                <1>       ;
                                <1>       ; 'syswait' waits for a process die.
                                <1>       ; It works in following way:
                                <1>       ;    1) From the parent process number, the parent's
                                <1>       ;      process name is found. The p.ppid table of parent
                                <1>       ;      names is then searched for this process name.
                                <1>       ;      If a match occurs, r2 contains child's process
                                <1>       ;      number. The child status is checked to see if it is
                                <1>       ;      a zombie, i.e; dead but not waited for (p.stat=3)
                                <1>       ;      If it is, the child process is freed and it's name
                                <1>       ;      is put in (u.r0). A return is then made via 'sysret'.
                                <1>       ;      If the child is not a zombie, nothing happens and
                                <1>       ;      the search goes on through the p.ppid table until
                                <1>       ;      all processes are checked or a zombie is found.
                                <1>       ;    2) If no zombies are found, a check is made to see if
                                <1>       ;      there are any children at all. If there are none,
                                <1>       ;      an error return is made. If there are, the parent's
                                <1>       ;      status is set to 2 (waiting for child to die),
                                <1>       ;      the parent is swapped out, and a branch to 'syswait'
                                <1>       ;      is made to wait on the next process.
                                <1>       ;
                                <1>       ; Calling sequence:
                                <1>       ;      ?
                                <1>       ; Arguments:
                                <1>       ;      -
                                <1>       ; Inputs: -
                                <1>       ; Outputs: if zombie found, it's name put in u.r0.
                                <1>       ; ..........................................................
                                <1>       ;
```

```
4000                                    <1>
4001                                    <1> ; / wait for a process to die
4002                                    <1>
4003                                    <1> syswait_0:
4004 0000D6AE 0FB61D[B3030300]          <1>       movzx  ebx, byte [u.uno] ; 01/09/2015
4005                                    <1>              ; movb u.uno,r1 / put parents process number in r1
4006 0000D6B5 D0E3                      <1>       shl    bl, 1
4007                                    <1>       ;shl   bx, 1
4008                                    <1>              ; asl r1 / x2 to get index into p.pid table
4009 0000D6B7 668B83[1E000300]          <1>       mov    ax, [ebx+p.pid-2]
4010                                    <1>              ; mov p.pid-2(r1),r1 / get the name of this process
4011 0000D6BE 31F6                      <1>       xor    esi, esi
4012                                    <1>              ; clr r2
4013 0000D6C0 31C9                      <1>       xor    ecx, ecx ; 30/10/2013
4014                                    <1>       ;xor   cl, cl
4015                                    <1>              ; clr r3 / initialize reg 3
4016                                    <1> syswait_1: ; 1:
4017 0000D6C2 6683C602                  <1>       add    si, 2
4018                                    <1>              ; add $2,r2 / use r2 for index into p.ppid table
4019                                    <1>                      ; / search table of parent processes
4020                                    <1>                      ; / for this process name
4021 0000D6C6 663B86[3E000300]          <1>       cmp    ax, [esi+p.ppid-2]
4022                                    <1>              ; cmp p.ppid-2(r2),r1 / r2 will contain the childs
4023                                    <1>                                   ; / process number
4024 0000D6CD 7535                      <1>       jne    short syswait_3
4025                                    <1>              ;bne 3f / branch if no match of parent process name
4026                                    <1>       ;inc   cx
4027 0000D6CF FEC1                      <1>       inc    cl
4028                                    <1>              ;inc r3 / yes, a match, r3 indicates number of children
4029 0000D6D1 66D1EE                    <1>       shr    si, 1
4030                                    <1>              ; asr r2 / r2/2 to get index to p.stat table
4031                                    <1>       ; The possible states ('p.stat' values) of a process are:
4032                                    <1>       ;      0 = free or unused
4033                                    <1>       ;      1 = active
4034                                    <1>       ;      2 = waiting for a child process to die
4035                                    <1>       ;      3 = terminated, but not yet waited for (zombie).
4036 0000D6D4 80BE[AF000300]03          <1>       cmp    byte [esi+p.stat-1], 3 ; SZOMB, 05/02/2014
4037                                    <1>              ; cmpb p.stat-1(r2),$3 / is the child process a zombie?
4038 0000D6DB 7524                      <1>       jne    short syswait_2
4039                                    <1>              ; bne 2f / no, skip it
4040 0000D6DD 88BE[AF000300]            <1>       mov    [esi+p.stat-1], bh ; 0
4041                                    <1>              ; clrb p.stat-1(r2) / yes, free it
4042 0000D6E3 66D1E6                    <1>       shl    si, 1
4043                                    <1>              ; asl r2 / r2x2 to get index into p.pid table
4044 0000D6E6 0FB786[1E000300]          <1>       movzx  eax, word [esi+p.pid-2]
4045 0000D6ED A3[64030300]              <1>       mov    [u.r0], eax
4046                                    <1>              ; mov p.pid-2(r2),*u.r0
4047                                    <1>                           ; / put childs process name in (u.r0)
4048                                    <1>       ;
4049                                    <1>       ; Retro UNIX 386 v1 modification ! (17/09/2015)
4050                                    <1>       ;
4051                                    <1>       ; Parent process ID -p.ppid- field (of the child process)
4052                                    <1>       ; must be cleared in order to prevent infinitive 'syswait'
4053                                    <1>       ; system call loop from the application/program if it calls
4054                                    <1>       ; 'syswait' again (mistakenly) while there is not a zombie
4055                                    <1>       ; or running child process to wait. ('forktest.s', 17/09/2015)
4056                                    <1>       ;
4057                                    <1>       ; Note: syswait will return with error if there is not a
4058                                    <1>       ;       zombie or running process to wait.
4059                                    <1>       ;
4060 0000D6F2 6629C0                    <1>       sub    ax, ax
4061 0000D6F5 668986[3E000300]          <1>       mov    [esi+p.ppid-2], ax ; 0 ; 17/09/2015
4062 0000D6FC E9D1FCFFFF                <1>       jmp    sysret0 ; ax = 0
4063                                    <1>       ;
4064                                    <1>       ;jmp   sysret
4065                                    <1>              ; br sysret1 / return cause child is dead
4066                                    <1> syswait_2: ; 2:
4067 0000D701 66D1E6                    <1>       shl    si, 1
4068                                    <1>              ; asl r2 / r2x2 to get index into p.ppid table
4069                                    <1> syswait_3: ; 3:
4070 0000D704 6683FE20                  <1>       cmp    si, nproc+nproc
4071                                    <1>              ; cmp r2,$nproc+nproc / have all processes been checked?
4072 0000D708 72B8                      <1>       jb     short syswait_1
4073                                    <1>              ; blt 1b / no, continue search
4074                                    <1>       ;and   cx, cx
4075 0000D70A 20C9                      <1>       and    cl, cl
4076                                    <1>              ; tst r3 / one gets here if there are no children
4077                                    <1>                     ; / or children that are still active
4078                                    <1>       ; 30/10/2013
4079 0000D70C 750B                      <1>       jnz    short syswait_4
4080                                    <1>       ;jz    error
4081                                    <1>              ; beq error1 / there are no children, error
4082 0000D70E 890D[64030300]            <1>       mov    [u.r0], ecx ; 0
4083 0000D714 E997FCFFFF                <1>       jmp    error
4084                                    <1> syswait_4:
4085 0000D719 8A1D[B3030300]            <1>       mov    bl, [u.uno]
4086                                    <1>              ; movb u.uno,r1 / there are children so put
4087                                    <1>                           ; / parent process number in r1
4088 0000D71F FE83[AF000300]            <1>       inc    byte [ebx+p.stat-1] ; 2, SWAIT, 05/02/2014
4089                                    <1>              ; incb p.stat-1(r1) / it is waiting for
4090                                    <1>                              ; / other children to die
4091                                    <1>       ; 04/11/2013
4092 0000D725 E83E3C0000                <1>       call   swap
4093                                    <1>              ; jsr r0,swap / swap it out, because it's waiting
4094 0000D72A EB82                      <1>       jmp    syswait_0
4095                                    <1>              ; br syswait / wait on next process
4096                                    <1>
4097                                    <1> sysfork: ; < create a new process >
4098                                    <1>       ; 02/01/2017 (TRDOS 386 modification)
4099                                    <1>       ; 04/09/2015, 18/05/2015
4100                                    <1>       ; 28/08/2015, 01/09/2015, 02/09/2015
4101                                    <1>       ; 09/05/2015, 10/05/2015, 14/05/2015
4102                                    <1>       ; 06/05/2015 (Retro UNIX 386 v1 - Beginning)
4103                                    <1>       ; 24/05/2013 - 14/02/2014 (Retro UNIX 8086 v1)
4104                                    <1>       ;
```

```
4105                                <1>       ; 'sysfork' creates a new process. This process is referred
4106                                <1>       ; to as the child process. This new process core image is
4107                                <1>       ; a copy of that of the caller of 'sysfork'. The only
4108                                <1>       ; distinction is the return location and the fact that (u.r0)
4109                                <1>       ; in the old process (parent) contains the process id (p.pid)
4110                                <1>       ; of the new process (child). This id is used by 'syswait'.
4111                                <1>       ; 'sysfork' works in the following manner:
4112                                <1>       ;    1) The process status table (p.stat) is searched to find
4113                                <1>       ;       a process number that is unused. If none are found
4114                                <1>       ;       an error occurs.
4115                                <1>       ;    2) when one is found, it becomes the child process number
4116                                <1>       ;       and it's status (p.stat) is set to active.
4117                                <1>       ;    3) If the parent had a control tty, the interrupt
4118                                <1>       ;       character in that tty buffer is cleared.
4119                                <1>       ;    4) The child process is put on the lowest priority run
4120                                <1>       ;       queue via 'putlu'.
4121                                <1>       ;    5) A new process name is gotten from 'mpid' (actually
4122                                <1>       ;       it is a unique number) and is put in the child's unique
4123                                <1>       ;       identifier; process id (p.pid).
4124                                <1>       ;    6) The process name of the parent is then obtained and
4125                                <1>       ;       placed in the unique identifier of the parent process
4126                                <1>       ;       name is then put in 'u.r0'.
4127                                <1>       ;    7) The child process is then written out on disk by
4128                                <1>       ;       'wswap',i.e., the parent process is copied onto disk
4129                                <1>       ;       and the child is born. (The child process is written
4130                                <1>       ;       out on disk/drum with 'u.uno' being the child process
4131                                <1>       ;       number.)
4132                                <1>       ;    8) The parent process number is then restored to 'u.uno'.
4133                                <1>       ;    9) The child process name is put in 'u.r0'.
4134                                <1>       ;   10) The pc on the stack sp + 18 is incremented by 2 to
4135                                <1>       ;       create the return address for the parent process.
4136                                <1>       ;   11) The 'u.fp' list as then searched to see what files
4137                                <1>       ;       the parent has opened. For each file the parent has
4138                                <1>       ;       opened, the corresponding 'fsp' entry must be updated
4139                                <1>       ;       to indicate that the child process also has opened
4140                                <1>       ;       the file. A branch to 'sysret' is then made.
4141                                <1>       ;
4142                                <1>       ; Calling sequence:
4143                                <1>       ;       from shell ?
4144                                <1>       ; Arguments:
4145                                <1>       ;       -
4146                                <1>       ; Inputs: -
4147                                <1>       ; Outputs: *u.r0 - child process name
4148                                <1>       ; ..........................................................
4149                                <1>       ;
4150                                <1>       ; Retro UNIX 8086 v1 modification:
4151                                <1>       ;       AX = r0 = PID (>0) (at the return of 'sysfork')
4152                                <1>       ;       = process id of child a parent process returns
4153                                <1>       ;       = process id of parent when a child process returns
4154                                <1>       ;
4155                                <1>       ;         In original UNIX v1, sysfork is called and returns as
4156                                <1>       ;       in following manner: (with an example: c library, fork)
4157                                <1>       ;
4158                                <1>       ;       1:
4159                                <1>       ;             sys   fork
4160                                <1>       ;                   br 1f  / child process returns here
4161                                <1>       ;             bes   2f     / parent process returns here
4162                                <1>       ;             / pid of new process in r0
4163                                <1>       ;             rts   pc
4164                                <1>       ;       2: / parent process condionally branches here
4165                                <1>       ;             mov   $-1,r0 / pid = -1 means error return
4166                                <1>       ;             rts   pc
4167                                <1>       ;
4168                                <1>       ;       1: / child process brances here
4169                                <1>       ;             clr   r0   / pid = 0 in child process
4170                                <1>       ;             rts   pc
4171                                <1>       ;
4172                                <1>       ;       In UNIX v7x86 (386) by Robert Nordier (1999)
4173                                <1>       ;             // pid = fork();
4174                                <1>       ;             //
4175                                <1>       ;             // pid == 0 in child process;
4176                                <1>       ;             // pid == -1 means error return
4177                                <1>       ;             // in child,
4178                                <1>       ;             //    parents id is in par_uid if needed
4179                                <1>       ;
4180                                <1>       ;             _fork:
4181                                <1>       ;                   mov   $.fork,eax
4182                                <1>       ;                   int   $0x30
4183                                <1>       ;                   jmp   1f
4184                                <1>       ;                   jnc   2f
4185                                <1>       ;                   jmp   cerror
4186                                <1>       ;             1:
4187                                <1>       ;                   mov   eax,_par_uid
4188                                <1>       ;                   xor   eax,eax
4189                                <1>       ;             2:
4190                                <1>       ;                   ret
4191                                <1>       ;
4192                                <1>       ;       In Retro UNIX 8086 v1,
4193                                <1>       ;       'sysfork' returns in following manner:
4194                                <1>       ;
4195                                <1>       ;             mov   ax, sys_fork
4196                                <1>       ;             mov   bx, offset @f ; routine for child
4197                                <1>       ;             int   20h
4198                                <1>       ;             jc    error
4199                                <1>       ;
4200                                <1>       ;       ; Routine for parent process here (just after 'jc')
4201                                <1>       ;             mov   word ptr [pid_of_child], ax
4202                                <1>       ;             jmp   next_routine_for_parent
4203                                <1>       ;
4204                                <1>       ;       @@: ; routine for child process here
4205                                <1>       ;             ....
4206                                <1>       ;       NOTE: 'sysfork' returns to specified offset
4207                                <1>       ;              for child process by using BX input.
4208                                <1>       ;             (at first, parent process will return then
4209                                <1>       ;             child process will return -after swapped in-
```

```
4210                                    <1>       ;           'syswait' is needed in parent process
4211                                    <1>       ;           if return from child process will be waited for.)
4212                                    <1>       ;
4213                                    <1>
4214                                    <1> ; / create a new process
4215                                    <1>       ; EBX = return address for child process
4216                                    <1>           ; (Retro UNIX 8086 v1 modification !)
4217 0000D72C 31F6                     <1>       xor   esi, esi
4218                                    <1>             ; clr r1
4219                                    <1> sysfork_1: ; 1: / search p.stat table for unused process number
4220 0000D72E 46                       <1>       inc   esi
4221                                    <1>             ; inc r1
4222 0000D72F 80BE[AF000300]00         <1>       cmp   byte [esi+p.stat-1], 0 ; SFREE, 05/02/2014
4223                                    <1>             ; tstb p.stat-1(r1) / is process active, unused, dead
4224 0000D736 760B                     <1>       jna   short sysfork_2
4225                                    <1>             ; beq 1f / it's unused so branch
4226 0000D738 6683FE10                 <1>       cmp   si, nproc
4227                                    <1>             ; cmp r1,$nproc / all processes checked
4228 0000D73C 72F0                     <1>       jb    short sysfork_1
4229                                    <1>             ; blt 1b / no, branch back
4230                                    <1>       ;
4231                                    <1>       ; Retro UNIX 8086 v1. modification:
4232                                    <1>       ;     Parent process returns from 'sysfork' to address
4233                                    <1>       ;     which is just after 'sysfork' system call in parent
4234                                    <1>       ;     process. Child process returns to address which is put
4235                                    <1>       ;     in BX register by parent process for 'sysfork'.
4236                                    <1>       ;
4237                                    <1>             ;add $2,18.(sp) / add 2 to pc when trap occured, points
4238                                    <1>                             ; / to old process return
4239                                    <1>             ; br error1 / no room for a new process
4240 0000D73E E96DFCFFFF               <1>       jmp   error
4241                                    <1> sysfork_2: ; 1:
4242 0000D743 E82D83FFFF               <1>       call  allocate_page
4243 0000D748 0F8262FCFFFF             <1>       jc    error
4244 0000D74E 50                       <1>       push  eax   ; UPAGE (user structure page) address
4245                                    <1>       ; Retro UNIX 386 v1 modification!
4246 0000D74F E81A85FFFF               <1>       call  duplicate_page_dir
4247                                    <1>             ; EAX = New page directory
4248 0000D754 730B                     <1>       jnc   short sysfork_3
4249 0000D756 58                       <1>       pop   eax   ; UPAGE (user structure page) address
4250 0000D757 E8E184FFFF               <1>       call  deallocate_page
4251 0000D75C E94FFCFFFF               <1>       jmp   error
4252                                    <1> sysfork_3:
4253                                    <1>       ; Retro UNIX 386 v1 modification !
4254 0000D761 56                       <1>       push  esi
4255 0000D762 E88F3C0000               <1>       call  wswap ; save current user (u) structure, user registers
4256                                    <1>                   ; and interrupt return components (for IRET)
4257 0000D767 8705[B8030300]           <1>       xchg  eax, [u.pgdir] ; page directory of the child process
4258 0000D76D A3[BC030300]             <1>       mov   [u.ppgdir], eax ; page directory of the parent process
4259 0000D772 5E                       <1>       pop   esi
4260 0000D773 58                       <1>       pop   eax   ; UPAGE (user structure page) address
4261                                    <1>             ; [u.usp] = esp
4262 0000D774 89F7                     <1>       mov   edi, esi
4263 0000D776 66C1E702                 <1>       shl   di, 2
4264 0000D77A 8987[BC000300]           <1>       mov   [edi+p.upage-4], eax ; memory page for 'user' struct
4265 0000D780 A3[B4030300]             <1>       mov   [u.upage], eax ; memory page for 'user' struct (child)
4266                                    <1>       ; 28/08/2015
4267 0000D785 0FB605[B3030300]         <1>       movzx eax, byte [u.uno] ; parent process number
4268                                    <1>             ; movb u.uno,-(sp) / save parent process number
4269 0000D78C 89C7                     <1>       mov   edi, eax
4270 0000D78E 50                       <1>       push  eax ; **
4271 0000D78F 8A87[7F000300]           <1>       mov    al, [edi+p.ttyc-1] ; console tty (parent)
4272                                    <1>       ; 18/09/2015
4273                                    <1>       ;mov    [esi+p.ttyc-1], al ; set child's console tty
4274                                    <1>       ;mov    [esi+p.waitc-1], ah ; 0 ; reset child's wait channel
4275 0000D795 668986[7F000300]         <1>       mov    [esi+p.ttyc-1], ax ; al - set child's console tty
4276                                    <1>                               ; ah - reset child's wait channel
4277 0000D79C 89F0                     <1>       mov   eax, esi
4278 0000D79E A2[B3030300]             <1>       mov   [u.uno], al ; child process number
4279                                    <1>             ;movb r1,u.uno / set child process number to r1
4280 0000D7A3 FE86[AF000300]           <1>         inc    byte [esi+p.stat-1] ; 1, SRUN, 05/02/2014
4281                                    <1>             ; incb p.stat-1(r1) / set p.stat entry for child
4282                                    <1>                           ; / process to active status
4283                                    <1>           ; mov u.ttyp,r2 / put pointer to parent process'
4284                                    <1>                         ; / control tty buffer in r2
4285                                    <1>               ; beq 2f / branch, if no such tty assigned
4286                                    <1>           ; clrb 6(r2) / clear interrupt character in tty buffer
4287                                    <1>       ; 2:
4288 0000D7A9 53                       <1>       push  ebx  ; * return address for the child process
4289                                    <1>                  ; * Retro UNIX 8086 v1 feature only !
4290                                    <1>       ; (Retro UNIX 8086 v1 modification!)
4291                                    <1>             ; mov $runq+4,r2
4292 0000D7AA BB[54030300]             <1>       mov   ebx, runq+2 ; normal run queue ; 02/01/2017
4293 0000D7AF E8B23C0000               <1>       call  putlu
4294                                    <1>             ; jsr r0,putlu / put child process on lowest priority
4295                                    <1>                       ; / run queue
4296 0000D7B4 66D1E6                   <1>       shl   si, 1
4297                                    <1>             ; asl r1 / multiply r1 by 2 to get index
4298                                    <1>                    ; / into p.pid table
4299 0000D7B7 66FF05[4E030300]         <1>       inc   word [mpid]
4300                                    <1>             ; inc mpid / increment m.pid; get a new process name
4301 0000D7BE 66A1[4E030300]           <1>       mov   ax, [mpid]
4302 0000D7C4 668986[1E000300]         <1>       mov   [esi+p.pid-2], ax
4303                                    <1>             ;mov mpid,p.pid-2(r1) / put new process name
4304                                    <1>                                   ; / in child process' name slot
4305 0000D7CB 5A                       <1>       pop   edx  ; * return address for the child process
4306                                    <1>                  ; * Retro UNIX 8086 v1 feature only !
4307 0000D7CC 5B                       <1>       pop   ebx  ; **
4308                                    <1>       ;mov  ebx, [esp] ; ** parent process number
4309                                    <1>             ; movb (sp),r2 / put parent process number in r2
4310 0000D7CD 66D1E3                   <1>       shl   bx, 1
4311                                    <1>             ;asl r2 / multiply by 2 to get index into below tables
4312                                    <1>       ;movzx eax, word [ebx+p.pid-2]
4313 0000D7D0 668B83[1E000300]         <1>       mov   ax, [ebx+p.pid-2]
4314                                    <1>             ; mov p.pid-2(r2),r2 / get process name of parent
```

```
4315                                    <1>                             ; / process
4316 0000D7D7 668986[3E000300]          <1>         mov   [esi+p.ppid-2], ax
4317                                    <1>               ; mov r2,p.ppid-2(r1) / put parent process name
4318                                    <1>                       ; / in parent process slot for child
4319 0000D7DE A3[64030300]              <1>         mov   [u.r0], eax
4320                                    <1>               ; mov r2,*u.r0 / put parent process name on stack
4321                                    <1>                           ; / at location where r0 was saved
4322 0000D7E3 8B2D[5C030300]            <1>         mov   ebp, [u.sp] ; points to return address (EIP for IRET)
4323 0000D7E9 895500                    <1>         mov   [ebp], edx ; *, CS:EIP -> EIP
4324                                    <1>                         ; * return address for the child process
4325                                    <1>               ; mov $sysret1,-(sp) /
4326                                    <1>               ; mov sp,u.usp / contents of sp at the time when
4327                                    <1>                            ; / user is swapped out
4328                                    <1>               ; mov $sstack,sp / point sp to swapping stack space
4329                                    <1>         ; 04/09/2015 - 01/09/2015
4330                                    <1>         ; [u.usp] = esp
4331 0000D7EC 68[D0D30000]              <1>         push  sysret ; ***
4332 0000D7F1 8925[60030300]            <1>         mov   [u.usp], esp ; points to 'sysret' address (***)
4333                                    <1>                           ; (for child process)
4334 0000D7F7 31C0                      <1>         xor   eax, eax
4335 0000D7F9 66A3[94030300]            <1>         mov   [u.ttyp], ax ; 0
4336                                    <1>         ;
4337 0000D7FF E8F23B0000                <1>         call  wswap ; Retro UNIX 8086 v1 modification !
4338                                    <1>               ;jsr r0,wswap / put child process out on drum
4339                                    <1>               ;jsr r0,unpack / unpack user stack
4340                                    <1>               ;mov u.usp,sp / restore user stack pointer
4341                                    <1>               ; tst (sp)+ / bump stack pointer
4342                                    <1>         ; Retro UNIX 386 v1 modification !
4343 0000D804 58                        <1>         pop   eax ; ***
4344 0000D805 66D1E3                    <1>         shl   bx, 1
4345 0000D808 8B83[BC000300]            <1>         mov   eax, [ebx+p.upage-4] ; UPAGE address ; 14/05/2015
4346 0000D80E E81B3C0000                <1>         call  rswap ; restore parent process 'u' structure,
4347                                    <1>                     ; registers and return address (for IRET)
4348                                    <1>               ;movb (sp)+,u.uno / put parent process number in u.uno
4349 0000D813 0FB705[4E030300]          <1>         movzx eax, word [mpid]
4350 0000D81A A3[64030300]              <1>         mov   [u.r0], eax
4351                                    <1>               ; mov mpid,*u.r0 / put child process name on stack
4352                                    <1>                              ; / where r0 was saved
4353                                    <1>               ; add $2,18.(sp) / add 2 to pc on stack; gives parent
4354                                    <1>                                ; / process return
4355                                    <1>         ;xor   ebx, ebx
4356 0000D81F 31F6                      <1>         xor   esi, esi
4357                                    <1>               ;clr r1
4358                                    <1> sysfork_4: ; 1: / search u.fp list to find the files
4359                                    <1>                    ; / opened by the parent process
4360                                    <1>         ; 01/09/2015
4361                                    <1>         ;xor   bh, bh
4362                                    <1>         ;mov   bl, [esi+u.fp]
4363 0000D821 8A86[6A030300]            <1>         mov   al, [esi+u.fp]
4364                                    <1>               ; movb u.fp(r1),r2 / get an open file for this process
4365                                    <1>         ;or      bl, bl
4366 0000D827 08C0                      <1>         or    al, al
4367 0000D829 740D                      <1>         jz    short sysfork_5
4368                                    <1>               ; beq 2f / file has not been opened by parent,
4369                                    <1>                      ; / so branch
4370 0000D82B B40A                      <1>         mov   ah, 10 ; Retro UNIX 386 v1 fsp structure size = 10 bytes
4371 0000D82D F6E4                      <1>         mul   ah
4372                                    <1>         ;movzx ebx, ax
4373 0000D82F 6689C3                    <1>         mov   bx, ax
4374                                    <1>         ;shl   bx, 3
4375                                    <1>               ; asl r2 / multiply by 8
4376                                    <1>                     ; asl r2 / to get index into fsp table
4377                                    <1>                     ; asl r2
4378 0000D832 FE83[4E010300]            <1>         inc   byte [ebx+fsp-2]
4379                                    <1>               ; incb fsp-2(r2) / increment number of processes
4380                                    <1>                           ; / using file, because child will now be
4381                                    <1>                           ; / using this file
4382                                    <1> sysfork_5: ; 2:
4383 0000D838 46                        <1>         inc   esi
4384                                    <1>               ; inc r1 / get next open file
4385 0000D839 6683FE0A                  <1>         cmp   si, 10
4386                                    <1>               ; cmp r1,$10. / 10. files is the maximum number which
4387                                    <1>                             ; / can be opened
4388 0000D83D 72E2                      <1>         jb    short sysfork_4
4389                                    <1>               ; blt 1b / check next entry
4390 0000D83F E98CFBFFFF                <1>         jmp   sysret
4391                                    <1>               ; br sysret1
4392                                    <1>
4393                                    <1> syscreat: ; < create file >
4394                                    <1>         ; 13/11/2017
4395                                    <1>         ; 27/10/2016
4396                                    <1>         ; 25/10/2016, 26/10/2016
4397                                    <1>         ; 15/10/2016, 16/10/2016, 17/10/2016
4398                                    <1>         ; 10/10/2016 (TRDOS 386 = TRDOS v2.0)
4399                                    <1>         ;           -derived from INT_21H.ASM-
4400                                    <1>         ;             ("loc_INT21h_create_file")
4401                                    <1>         ;      10/07/2011 (12/03/2011)
4402                                    <1>         ;      INT 21h Function AH = 3Ch
4403                                    <1>         ;      Create File
4404                                    <1>         ;      INPUT
4405                                    <1>         ;         CX = Attributes
4406                                    <1>         ;           DS:DX= Address of zero terminaned path name
4407                                    <1>         ;
4408                                    <1>         ; 27/12/2015 (Retro UNIX 386 v1.1)
4409                                    <1>         ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
4410                                    <1>         ; 27/05/2013 (Retro UNIX 8086 v1)
4411                                    <1>         ;
4412                                    <1>         ; 'syscreat' called with two arguments; name and mode.
4413                                    <1>         ; u.namep points to name of the file and mode is put
4414                                    <1>         ; on the stack. 'namei' is called to get i-number of the file.
4415                                    <1>         ; If the file aready exists, it's mode and owner remain
4416                                    <1>         ; unchanged, but it is truncated to zero length. If the file
4417                                    <1>         ; did not exist, an i-node is created with the new mode via
4418                                    <1>         ; 'maknod' whether or not the file already existed, it is
4419                                    <1>         ; open for writing. The fsp table is then searched for a free
```

```
4420                                 <1>       ; entry. When a free entry is found, proper data is placed
4421                                 <1>       ; in it and the number of this entry is put in the u.fp list.
4422                                 <1>       ; The index to the u.fp (also know as the file descriptor)
4423                                 <1>       ; is put in the user's r0.
4424                                 <1>       ;
4425                                 <1>       ; Calling sequence:
4426                                 <1>       ;     syscreate; name; mode
4427                                 <1>       ; Arguments:
4428                                 <1>       ;     name - name of the file to be created
4429                                 <1>       ;     mode - mode of the file to be created
4430                                 <1>       ; Inputs: (arguments)
4431                                 <1>       ; Outputs: *u.r0 - index to u.fp list
4432                                 <1>       ;               (the file descriptor of new file)
4433                                 <1>       ; ..........................................................
4434                                 <1>       ;
4435                                 <1>       ; Retro UNIX 8086 v1 modification:
4436                                 <1>       ;       'syscreate' system call has two arguments; so,
4437                                 <1>       ;     * 1st argument, name is pointed to by BX register
4438                                 <1>       ;     * 2nd argument, mode is in CX register
4439                                 <1>       ;
4440                                 <1>       ;     AX register (will be restored via 'u.r0') will return
4441                                 <1>       ;     to the user with the file descriptor/number
4442                                 <1>       ;     (index to u.fp list).
4443                                 <1>       ;
4444                                 <1>       ;call  arg2
4445                                 <1>       ; * name - 'u.namep' points to address of file/path name
4446                                 <1>       ;          in the user's program segment ('u.segmnt')
4447                                 <1>       ;          with offset in BX register (as sysopen argument 1).
4448                                 <1>       ; * mode - sysopen argument 2 is in CX register
4449                                 <1>       ;          which is on top of stack.
4450                                 <1>       ;
4451                                 <1>       ; TRDOS 386 (10/10/2016)
4452                                 <1>       ;
4453                                 <1>       ; INPUT ->
4454                                 <1>       ;        CL = File Attributes
4455                                 <1>       ;           bit 0 (1) - Read only file (R)
4456                                 <1>       ;             bit 1 (1) - Hidden file (H)
4457                                 <1>       ;             bit 2 (1) - System file (R)
4458                                 <1>       ;             bit 3 (1) - Volume label/name (V)
4459                                 <1>       ;             bit 4 (1) - Subdirectory (D)
4460                                 <1>       ;           bit 5 (1) - File has been archived (A)
4461                                 <1>       ;          EBX = Pointer to filename (ASCIIZ) -path-
4462                                 <1>       ;
4463                                 <1>       ; OUTPUT ->
4464                                 <1>       ;          eax = File/Device Handle/Number (index) (AL)
4465                                 <1>       ;          cf = 1 -> Error code in AL
4466                                 <1>       ;
4467                                 <1>       ; Modified Registers: EAX (at the return of system call)
4468                                 <1>       ;
4469                                 <1>       ; Note: If the file is existing and it has not any one
4470                                 <1>       ;     of S,H,R,V,D attributes, it will be truncated
4471                                 <1>       ;     to zero length; otherwise, access error will be
4472                                 <1>       ;     returned.
4473                                 <1>
4474                                 <1> sysmkdir_0:
4475 0000D844 F6C108              <1>       test   cl, 08h ; Volume name
4476 0000D847 740A                  <1>       jz     short syscreat_0
4477                                 <1>
4478                                 <1>       ; Volume name or long name creation
4479                                 <1>       ; is not permitted (in TRDOS 386)!
4480 0000D849 B80B000000            <1>       mov    eax, ERR_FILE_ACCESS  ; 11 ; 'permission denied !'
4481 0000D84E E9CC000000            <1>         jmp  sysopen_dev_err
4482                                 <1>
4483                                 <1> syscreat_0:
4484                                 <1>         ;mov [u.namep], ebx
4485 0000D853 51                    <1>       push   ecx
4486 0000D854 89DE                  <1>       mov    esi, ebx
4487                                 <1>       ; file name is forced, change directory as temporary
4488                                 <1>       ;mov   ax, 1
4489                                 <1>       ;mov   [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
4490                                 <1>       ;call  set_working_path
4491 0000D856 E863510000            <1>       call   set_working_path_x ; 17/10/2016
4492 0000D85B 0F82D7000000          <1>       jc     syscreat_err
4493                                 <1>
4494                                 <1>       ; 16/10/2016
4495 0000D861 803D[5F8D0100]00      <1>       cmp    byte [SWP_inv_fname], 0
4496 0000D868 776C                  <1>       ja     short  syscreat_inv_fname ; invalid file name !
4497                                 <1>
4498                                 <1>       ; Here, we have a valid path and also a valid file name
4499                                 <1>       ; (Working dir has been changed if the path
4500                                 <1>       ;  -file name string- had contained a dir name.)
4501                                 <1>
4502 0000D86A 6631C0                <1>       xor    ax, ax
4503                                 <1>       ;mov   esi, FindFile_Name
4504 0000D86D E8E3B6FFFF            <1>       call   find_first_file
4505 0000D872 59                    <1>       pop    ecx
4506                                 <1>              ; ESI = Directory Entry (FindFile_DirEntry) Location
4507                                 <1>              ; EDI = Directory Buffer Directory Entry Location
4508                                 <1>              ; EAX = File Size
4509                                 <1>              ;  BL = Attributes of The File/Directory
4510                                 <1>              ;  BH = Long Name Yes/No Status (>0 is YES)
4511                                 <1>              ;  DX > 0 : Ambiguous filename chars are used
4512 0000D873 7269                  <1>       jc     short syscreat_1 ; file not found (the good!)
4513                                 <1>                             ; or another error (the bad')
4514                                 <1>
4515                                 <1>       ; (& the uggly!) truncate file to zero length before open
4516                                 <1>
4517                                 <1>       ;'*' and '?' already checked at 'set_working_path' stage
4518                                 <1>       ;and   dx, dx
4519                                 <1>       ;jnz   short sysmkdir_err ; permission denied
4520                                 <1>                              ; invalid filename chars
4521                                 <1>
4522                                 <1>       ;test  cl, 10h ; subdirectory ?
4523                                 <1>       ;jnz   short sysmkdir_err
4524                                 <1>
```

```
4525                                    <1>      ; BL = File Attributes:
4526                                    <1>      ;            bit 0 (1) - Read only file (R)
4527                                    <1>      ;              bit 1 (1) - Hidden file (H)
4528                                    <1>        ;              bit 2 (1) - System file (R)
4529                                    <1>      ;              bit 3 (1) - Volume label/name (V)
4530                                    <1>        ;              bit 4 (1) - Subdirectory (D)
4531                                    <1>      ;            bit 5 (1) - File has been archived
4532                                    <1>
4533                                    <1>      ; * existing directory must not be truncated
4534                                    <1>      ;   (we don't know it is empty or not, at this stage)
4535                                    <1>      ; * existing volume name (or a long name) can not be
4536                                    <1>      ;   re-created or truncated by 'syscreat'
4537                                    <1>      ; * A file with S, H, R attributes must not be truncated
4538                                    <1>      ;   (change attributes to normal, if you need truncate it)
4539                                    <1>
4540 0000D875 F6C31F                  <1>      test   bl, 00011111b  ; check attributes of existing file
4541 0000D878 754E                     <1>      jnz    short sysmkdir_err
4542                                    <1>
4543                                    <1>      ;; normal file, OK to continue...
4544                                    <1>
4545                                    <1>      ; ESI = FindFile_DirEntry
4546 0000D87A 668B4614                 <1>      mov    ax, [esi+DirEntry_FstClusHI] ; 20
4547 0000D87E C1E010                   <1>      shl    eax, 16 ; 13/11/2017
4548 0000D881 668B461A                 <1>      mov    ax, [esi+DirEntry_FstClusLO] ; 26
4549                                    <1>      ; EAX = First cluster to be truncated/unlinked
4550 0000D885 57                       <1>      push   edi
4551 0000D886 51                       <1>      push   ecx
4552 0000D887 BE00010900               <1>      mov    esi, Logical_DOSDisks
4553 0000D88C 29C9                     <1>      sub    ecx, ecx
4554 0000D88E 8A2D[6E810100]           <1>      mov    ch, [Current_Drv]
4555 0000D894 01CE                     <1>      add    esi, ecx
4556                                    <1>      ; ESI = Logical dos drive description table address
4557 0000D896 E8C9F7FFFF               <1>      call   truncate_cluster_chain
4558 0000D89B 59                       <1>      pop    ecx
4559 0000D89C 5F                       <1>      pop    edi
4560 0000D89D 7230                     <1>      jc     short syscreate_truncate_err
4561                                    <1>
4562                                    <1>      ; 26/10/2016
4563                                    <1>      ; EDI = Directory entry address in directory buffer
4564                                    <1>      ; Update directory entry
4565 0000D89F E848DCFFFF               <1>      call   convert_current_date_time
4566                                    <1>      ; OUTPUT -> DX = Date in dos dir entry format
4567                                    <1>        ;       AX = Time in dos dir entry format
4568 0000D8A4 66894716                 <1>      mov    [edi+DirEntry_WrtTime], ax
4569 0000D8A8 66895718                 <1>      mov    [edi+DirEntry_WrtDate], dx
4570 0000D8AC 66895712                 <1>      mov    [edi+DirEntry_LastAccDate], dx
4571 0000D8B0 31C0                     <1>      xor    eax, eax ; file size = 0
4572 0000D8B2 89471C                   <1>      mov    [edi+DirEntry_FileSize], eax ; 0
4573 0000D8B5 C605[94880100]02         <1>      mov    byte [DirBuff_ValidData], 2 ; data changed sign
4574 0000D8BC BE[608A0100]             <1>      mov    esi, FindFile_DirEntry
4575 0000D8C1 B201                     <1>      mov    dl, 1 ; open file for writing
4576 0000D8C3 E9AA000000               <1>      jmp    sysopen_2
4577                                    <1>
4578                                    <1> sysmkdir_err:
4579                                    <1>      ; 1 = write, 2 = read & write, >2 = invalid
4580 0000D8C8 B80B000000               <1>      mov    eax, ERR_FILE_ACCESS  ; 11 ; 'permission denied !'
4581 0000D8CD EB73                     <1>      jmp    short sysopen_err
4582                                    <1>
4583                                    <1> syscreate_truncate_err:
4584 0000D8CF B812000000               <1>      mov    eax, ERR_DRV_WRITE ; 18 ; 'disk write error !'
4585 0000D8D4 EB6C                     <1>      jmp    short sysopen_err
4586                                    <1>
4587                                    <1> syscreat_inv_fname:  ; invalid file name chars
4588                                    <1>      ; 16/10/2016
4589 0000D8D6 B81A000000               <1>      mov    eax, ERR_INV_FILE_NAME  ; 26 ; invalid file name chars
4590 0000D8DB 59                       <1>      pop    ecx
4591 0000D8DC EB64                     <1>      jmp    sysopen_err
4592                                    <1>
4593                                    <1> syscreat_1:
4594                                    <1>      ; Error code in EAX
4595 0000D8DE 3C02                     <1>      cmp    al, 02h ; 'File not found' error
4596 0000D8E0 7560                     <1>      jne    sysopen_err
4597                                    <1>
4598 0000D8E2 F6C110                   <1>      test   cl, 10h ; Directory
4599 0000D8E5 0F8597010000             <1>      jnz    sysmkdir_2
4600                                    <1>
4601                                    <1> syscreat_2:
4602 0000D8EB BE[508A0100]             <1>      mov    esi, FindFile_Name
4603                                    <1>      ;xor   edx, edx
4604 0000D8F0 31C0                     <1>      xor    eax, eax ; File Size  = 0
4605 0000D8F2 31DB                     <1>      xor    ebx, ebx
4606 0000D8F4 4B                       <1>      dec    ebx ; FFFFFFFFh -> create empty file
4607                                    <1>                    ;              (only for FAT fs)
4608                                    <1>      ; CL = File Attributes
4609 0000D8F5 E8F8EBFFFF               <1>      call   create_file
4610 0000D8FA 7246                     <1>      jc     sysopen_err
4611                                    <1>             ; EAX = New file's first cluster
4612                                    <1>             ; ESI = Logical Dos Drv Descr. Table Addr.
4613                                    <1>             ; EBX = offset CreateFile_Size
4614                                    <1>             ; ECX = Sectors per cluster (<256)
4615                                    <1>             ; EDX = Directory entry index/number (<65536)
4616                                    <1>      ; 26/10/2016
4617                                    <1>      ;mov   esi, Directory_Buffer
4618                                    <1>      ;shl   dx, 5 ; *32
4619                                    <1>      ;add   esi, edx
4620                                    <1>      ;; esi = directory entry address in directory buffer
4621                                    <1>
4622                                    <1>      ; Here, directory entry has been created but last
4623                                    <1>      ; modification date & time of the parent dir has not
4624                                    <1>      ; been updated, yet!
4625                                    <1>      ; (Note: Directory and FAT buffers have been updated...)
4626                                    <1>
4627 0000D8FC E824DDFFFF               <1>      call   update_parent_dir_lmdt ; now, it is OK too!
4628                                    <1>
4629                                    <1>      ; 25/10/2016
```

```
4630 0000D901 66B80018            <1>       mov    ax, 1800h
4631 0000D905 BE[508A0100]        <1>       mov    esi, FindFile_Name
4632 0000D90A E846B6FFFF          <1>       call   find_first_file
4633 0000D90F 7231                <1>       jc     short sysopen_err
4634                              <1>
4635                              <1>       ; Only possible error after here is
4636                              <1>       ; "too many open files !" error.
4637                              <1>       ;
4638                              <1>       ; If "syscreat" will return with that error,
4639                              <1>       ; (the file has been created but it could not be opened)
4640                              <1>       ; the user must retry to open this file again
4641                              <1>       ; or must close another file before using
4642                              <1>       ; "sysopen" system call.
4643                              <1>
4644 0000D911 B201                <1>       mov    dl, 1 ; open file for writing
4645                              <1>       ; ESI = Directory Entry (FindFile_DirEntry) Location
4646                              <1>       ; EAX = File Size (= 0)
4647 0000D913 EB5D                <1>       jmp    short sysopen_2
4648                              <1>
4649                              <1> sysopen: ;<open file>
4650                              <1>       ; 17/04/2021 - TRDOS 386 v2.0.4
4651                              <1>       ;      (temporary modifications)
4652                              <1>       ; 26/10/2016
4653                              <1>       ; 24/10/2016
4654                              <1>       ; 17/10/2016
4655                              <1>       ; 15/10/2016
4656                              <1>       ; 06/10/2016, 07/10/2016, 08/10/2016
4657                              <1>       ; 05/10/2016 (TRDOS 386 = TRDOS v2.0)
4658                              <1>       ;           -derived from INT_21H.ASM-
4659                              <1>       ;             ("loc_INT21h_open_file")
4660                              <1>       ;      26/02/2011
4661                              <1>       ;      INT 21h Function AH = 3Dh
4662                              <1>       ;      Open File
4663                              <1>       ;      INPUT
4664                              <1>       ;         AL= File Access Value
4665                              <1>       ;           0- Open for reading
4666                              <1>       ;             1- Open for writing
4667                              <1>       ;             2- Open for reading and writing
4668                              <1>       ;          DS:DX= Pointer to filename (ASCIIZ)
4669                              <1>       ;
4670                              <1>       ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
4671                              <1>       ; 22/05/2013 - 27/05/2013 (Retro UNIX 8086 v1)
4672                              <1>       ;
4673                              <1>       ; 'sysopen' opens a file in following manner:
4674                              <1>       ;    1) The second argument in a sysopen says whether to
4675                              <1>       ;      open the file ro read (0) or write (>0).
4676                              <1>       ;    2) I-node of the particular file is obtained via 'namei'.
4677                              <1>       ;    3) The file is opened by 'iopen'.
4678                              <1>       ;    4) Next housekeeping is performed on the fsp table
4679                              <1>       ;      and the user's open file list - u.fp.
4680                              <1>       ;      a) u.fp and fsp are scanned for the next available slot.
4681                              <1>       ;      b) An entry for the file is created in the fsp table.
4682                              <1>       ;      c) The number of this entry is put on u.fp list.
4683                              <1>       ;      d) The file descriptor index to u.fp list is pointed
4684                              <1>       ;         to by u.r0.
4685                              <1>       ;
4686                              <1>       ; Calling sequence:
4687                              <1>       ;      sysopen; name; mode
4688                              <1>       ; Arguments:
4689                              <1>       ;      name - file name or path name
4690                              <1>       ;      mode - 0 to open for reading
4691                              <1>       ;             1 to open for writing
4692                              <1>       ; Inputs: (arguments)
4693                              <1>       ; Outputs: *u.r0 - index to u.fp list (the file descriptor)
4694                              <1>       ;              is put into r0's location on the stack.
4695                              <1>       ; ..........................................................
4696                              <1>       ;
4697                              <1>       ; Retro UNIX 8086 v1 modification:
4698                              <1>       ;       'sysopen' system call has two arguments; so,
4699                              <1>       ;      * 1st argument, name is pointed to by BX register
4700                              <1>       ;      * 2nd argument, mode is in CX register
4701                              <1>       ;
4702                              <1>       ;      AX register (will be restored via 'u.r0') will return
4703                              <1>       ;      to the user with the file descriptor/number
4704                              <1>       ;      (index to u.fp list).
4705                              <1>       ;
4706                              <1>       ;call  arg2
4707                              <1>       ; * name - 'u.namep' points to address of file/path name
4708                              <1>       ;          in the user's program segment ('u.segmnt')
4709                              <1>       ;          with offset in BX register (as sysopen argument 1).
4710                              <1>       ; * mode - sysopen argument 2 is in CX register
4711                              <1>       ;          which is on top of stack.
4712                              <1>       ;
4713                              <1>       ; jsr r0,arg2 / get sys args into u.namep and on stack
4714                              <1>       ;
4715                              <1>       ; system call registers: ebx, ecx (through 'sysenter')
4716                              <1>       ;
4717                              <1>       ; TRDOS 386 (05/10/2016)
4718                              <1>       ;
4719                              <1>       ; INPUT ->
4720                              <1>       ;        CL = File Access Value (Open Mode)
4721                              <1>       ;           0 - Open file for reading
4722                              <1>       ;             1 - Open file for writing
4723                              <1>       ;             2 - Open device for reading
4724                              <1>       ;           3 - Open device for writing
4725                              <1>       ;          EBX = Pointer to filename/devicename (ASCIIZ)
4726                              <1>       ; OUTPUT ->
4727                              <1>       ;          eax = File/Device Handle/Number (index) (AL)
4728                              <1>       ;          cf = 1 -> Error code in AL
4729                              <1>       ;
4730                              <1>       ; Modified Registers: EAX (at the return of system call)
4731                              <1>       ;
4732                              <1>
4733 0000D915 80F901              <1>       cmp    cl, 1 ; read file (0), write file (1)
4734 0000D918 7614                <1>       jna    short sysopen_0
```

```
                                <1>
                                <1>       ; 17/04/2021 (temporary)
                                <1>       ;cmp   cl, 3
                                <1>       ;jna   sysopen_device
                                <1>
                                <1>       ; Invalid access code
0000D91A B817000000             <1>       mov    eax, ERR_INV_PARAMETER
                                <1>       ;jmp   sysopen_dev_err
                                <1>
                                <1> sysopen_dev_err:
0000D91F A3[64030300]           <1>       mov    [u.r0], eax
0000D924 A3[C8030300]           <1>       mov    [u.error], eax
0000D929 E982FAFFFF             <1>       jmp    error
                                <1>
                                <1> sysopen_0:
                                <1>       ;mov   [u.namep], ebx
0000D92E 51                     <1>       push   ecx
0000D92F 89DE                   <1>       mov    esi, ebx
                                <1>       ; file name is forced, change directory as temporary
                                <1>       ;mov   ax, 1
                                <1>       ;mov   [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
                                <1>       ;call  set_working_path
0000D931 E888500000             <1>       call   set_working_path_x ; 17/10/2016
0000D936 731E                   <1>       jnc    short sysopen_1
                                <1>
                                <1> syscreat_err: ; ecx = file attributes (for 'syscreat')
0000D938 59                     <1>       pop    ecx ; open mode
0000D939 21C0                   <1>       and    eax, eax  ; 0 -> Bad Path!
0000D93B 7505                   <1>       jnz    short sysopen_err
                                <1>       ; eax = 0
0000D93D B80C000000             <1>       mov    eax, ERR_DIR_NOT_FOUND ; Directory not found !
                                <1> sysopen_err:
0000D942 A3[64030300]           <1>       mov    [u.r0], eax
0000D947 A3[C8030300]           <1>       mov    [u.error], eax
0000D94C E842510000             <1>       call   reset_working_path
0000D951 E95AFAFFFF             <1>       jmp    error
                                <1>
                                <1> sysopen_1:
                                <1>       ;mov   esi, FindFile_Name
0000D956 66B80018               <1>       mov    ax, 1800h ; Only files
0000D95A E8F6B5FFFF             <1>       call   find_first_file
0000D95F 5A                     <1>       pop    edx
0000D960 72E0                   <1>       jc     short sysopen_err ; eax = 2 (File not found !)
                                <1>
                                <1>       ; check_open_file_attr_access_code
                                <1>
0000D962 F6C307                 <1>       test   bl, 7  ; system, hidden, readonly
0000D965 740B                   <1>       jz     short sysopen_2
                                <1>
0000D967 20D2                   <1>       and    dl, dl ; 0 = read mode
0000D969 7407                   <1>       jz     short sysopen_2
                                <1>
                                <1>       ; 1 = write, 2 = read & write, >2 = invalid
0000D96B B80B000000             <1>       mov    eax, ERR_FILE_ACCESS ; 11 = 'permission denied !'
0000D970 EBD0                   <1>       jmp    short sysopen_err
                                <1>
                                <1> sysopen_2:
                                <1>       ; esi = Directory Entry (FindFile_DirEntry) Location
0000D972 89F3                   <1>       mov    ebx, esi
0000D974 31F6                   <1>       xor    esi, esi ; 0
0000D976 31FF                   <1>       xor    edi, edi ; 0
                                <1> sysopen_3: ; scan the list of entries in fsp table
0000D978 80BE[6A030300]00       <1>       cmp    byte [esi+u.fp], 0
0000D97F 760F                   <1>       jna    short sysopen_4 ; empty slot
0000D981 6646                   <1>       inc    si
0000D983 6683FE0A               <1>       cmp    si, 10
0000D987 72EF                   <1>       jb     short sysopen_3
                                <1> toomanyf:
0000D989 B80D000000             <1>       mov    eax, ERR_TOO_MANY_FILES ; too many open files !
0000D98E EBB2                   <1>       jmp    short sysopen_err
                                <1>
                                <1> sysopen_4:
0000D990 80BF[9A8D0100]00       <1>       cmp    byte [edi+OF_MODE], 0 ; Scan open files table
0000D997 760A                   <1>       jna    short sysopen_5
0000D999 6647                   <1>       inc    di
0000D99B 6683FF0A               <1>       cmp    di, OPENFILES ; max. number of open files (=10)
0000D99F 72EF                   <1>       jb     short sysopen_4
0000D9A1 EBE6                   <1>       jmp    short toomanyf
                                <1>
                                <1> sysopen_5:
0000D9A3 FEC2                   <1>       inc    dl
0000D9A5 8897[9A8D0100]         <1>       mov    [edi+OF_MODE], dl
0000D9AB 8A15[0E8A0100]         <1>       mov    dl, [FindFile_Drv]
0000D9B1 8897[908D0100]         <1>       mov    [edi+OF_DRIVE], dl ; Logical DOS drive number
0000D9B7 66C1E702               <1>       shl    di, 2 ; *4 (dword offset)
                                <1>
0000D9BB 8987[E08D0100]         <1>       mov    [edi+OF_SIZE], eax ; File size in bytes
                                <1>
0000D9C1 668B4314               <1>       mov    ax, [ebx+DirEntry_FstClusHI]
0000D9C5 C1E010                 <1>       shl    eax, 16
0000D9C8 668B431A               <1>       mov    ax, [ebx+DirEntry_FstClusLO]
0000D9CC 8987[688D0100]         <1>       mov    [edi+OF_FCLUSTER], eax ; First cluster
0000D9D2 8987[808E0100]         <1>       mov    [edi+OF_CCLUSTER], eax ; Current cluster
                                <1>
0000D9D8 31DB                   <1>       xor    ebx, ebx
0000D9DA 899F[B88D0100]         <1>       mov    [edi+OF_POINTER], ebx ; offset pointer (0)
0000D9E0 899F[A88E0100]         <1>       mov    [edi+OF_CCINDEX], ebx ; cluster index (0)
                                <1>
0000D9E6 A1[808A0100]           <1>       mov    eax, [FindFile_DirFirstCluster]
0000D9EB 8987[088E0100]         <1>       mov    [edi+OF_DIRFCLUSTER], eax
                                <1>
0000D9F1 A1[848A0100]           <1>       mov    eax, [FindFile_DirCluster]
0000D9F6 8987[308E0100]         <1>       mov    [edi+OF_DIRCLUSTER], eax
                                <1>
                                <1>       ; Get (& Save) Volume ID
```

```
4840                            <1>       ; Important for files of removable drives
4841                            <1>       ; (In order to check the drive has same volume/disk)
4842 0000D9FC 88D7              <1>       mov    bh, dl
4843 0000D9FE 81C300010900      <1>       add    ebx, Logical_DOSDisks
4844 0000DA04 8A4303            <1>       mov    al, [ebx+LD_FATType]
4845 0000DA07 3C01              <1>       cmp    al, 1
4846 0000DA09 7209              <1>       jb     short sysopen_6_fs
4847 0000DA0B 3C02              <1>       cmp    al, 2
4848 0000DA0D 770A              <1>       ja     short sysopen_6_fat32
4849                            <1> sysopen_6_fat:
4850 0000DA0F 8B432D            <1>       mov    eax, [ebx+LD_BPB+VolumeID]
4851 0000DA12 EB08              <1>       jmp    short sysopen_7
4852                            <1> sysopen_6_fs:
4853 0000DA14 8B4328            <1>       mov    eax, [ebx+LD_FS_VolumeSerial]
4854 0000DA17 EB03              <1>       jmp    short sysopen_7
4855                            <1> sysopen_6_fat32:
4856 0000DA19 8B4349            <1>       mov    eax, [ebx+LD_BPB+FAT32_VolID]
4857                            <1> sysopen_7:
4858 0000DA1C A3[64810100]      <1>       mov    [Current_VolSerial], eax
4859                            <1>
4860 0000DA21 8987[588E0100]    <1>       mov    [edi+OF_VOLUMEID], eax
4861                            <1>
4862                            <1>       ; 24/10/2016
4863 0000DA27 66D1EF            <1>       shr    di, 1 ; 4/2, word offset
4864 0000DA2A 668B1D[888A0100]  <1>       mov    bx, [FindFile_DirEntryNumber]
4865 0000DA31 66899F[D08E0100]  <1>       mov    [edi+OF_DIRENTRY], bx
4866                            <1>
4867 0000DA38 31D2              <1>       xor    edx, edx
4868                            <1>       ;shr   di, 2 ; /4 (byte offset)
4869 0000DA3A 66D1EF            <1>       shr    di, 1 ; 2/2, byte offset
4870 0000DA3D 8897[AE8D0100]    <1>       mov    byte [edi+OF_OPENCOUNT], dl ; 0
4871 0000DA43 8897[A48D0100]    <1>       mov    byte [edi+OF_STATUS], dl ; 0
4872                            <1>
4873 0000DA49 89FB              <1>       mov    ebx, edi
4874 0000DA4B FEC3              <1>       inc    bl
4875                            <1>
4876 0000DA4D 889E[6A030300]    <1>       mov   [esi+u.fp], bl ; Open File Entry Number
4877 0000DA53 8935[64030300]    <1>       mov   [u.r0], esi ; move index to u.fp list
4878                            <1>                         ; into eax on stack
4879                            <1>
4880 0000DA59 E835500000        <1>       call   reset_working_path
4881                            <1>
4882 0000DA5E E96DF9FFFF        <1>       jmp    sysret
4883                            <1>
4884                            <1>       ; (Retro UNIX 386 v1.0)
4885                            <1>       ; 'fsp' table (10 bytes/entry)
4886                            <1>       ; bit 15                              bit 0
4887                            <1>       ; ---|----------------------------------------
4888                            <1>       ; r/w|       i-number of open file
4889                            <1>       ; ---|----------------------------------------
4890                            <1>       ;               device number
4891                            <1>       ; ----------------------------------------------
4892                            <1>       ; offset pointer, r/w pointer to file (bit 0-15)
4893                            <1>       ; ----------------------------------------------
4894                            <1>       ; offset pointer, r/w pointer to file (bit 16-31)
4895                            <1>       ; ----------------------|-----------------------
4896                            <1>       ;  flag that says file    | number of processes
4897                            <1>       ;   has been deleted      | that have file open
4898                            <1>       ; ----------------------|-----------------------
4899                            <1>
4900                            <1> ; 17/04/2021
4901                            <1> ; ('sysopen_device' procedure is disabled as temporary)
4902                            <1>
4903                            <1> ;sysopen_device:
4904                            <1> ;     ; 15/10/2016
4905                            <1> ;     ; 08/10/2016
4906                            <1> ;     ; 07/10/2016 (TRDOS 386 = TRDOS v2.0)
4907                            <1> ;     push   ecx ; open mode
4908                            <1> ;     mov    ebp, esp
4909                            <1> ;     mov    ecx, 16 ; transfer length = 16 bytes
4910                            <1> ;     sub    esp, ecx
4911                            <1> ;     mov    edi, esp ; destination address
4912                            <1> ;     mov    esi, ebx ; dev name in user's memory space
4913                            <1> ;     call   transfer_from_user_buffer
4914                            <1> ;     jnc    short sysopen_dev_0
4915                            <1> ;     ; eax = ERR_OUT_OF_MEMORY = 4 = ERR_MINOR_IM
4916                            <1> ;     pop    ecx
4917                            <1> ;sysopen_dev_err:
4918                            <1> ;     mov    [u.r0], eax
4919                            <1> ;     mov    [u.error], eax
4920                            <1> ;     jmp    error
4921                            <1> ;sysopen_dev_0:
4922                            <1> ;     mov    esi, edi ; Device name addr (max. 16 bytes, ASCIIZ)
4923                            <1> ;                     ; for example: "tty, TTY, /dev/tty"
4924                            <1> ;     call   get_device_number
4925                            <1> ;     mov    esp, ebp
4926                            <1> ;     pop    ecx
4927                            <1> ;     jnc    short sysopen_dev_1
4928                            <1> ;     mov    eax, ERR_INV_DEV_NAME ; 24 ; 'invalid device name !'
4929                            <1> ;     jmp    short sysopen_dev_err
4930                            <1> ;sysopen_dev_1:
4931                            <1> ;     ; eax = Device Number (AL)
4932                            <1> ;     ;  cl = Open mode (2 = device read, 3 = device write)
4933                            <1> ;       xor     ebx, ebx ; 0
4934                            <1> ;sysopen_dev_2: ; scan the list of entries
4935                            <1> ;       cmp     [ebx+u.fp], bl ; 0
4936                            <1> ;       jna     short sysopen_dev_3 ; empty slot
4937                            <1> ;       inc     bl
4938                            <1> ;       cmp     bl, 10
4939                            <1> ;     jb     short sysopen_dev_2
4940                            <1> ;     ;
4941                            <1> ;     mov    eax, ERR_TOO_MANY_FILES ; too many open files !
4942                            <1> ;     jmp    short sysopen_dev_err
4943                            <1> ;sysopen_dev_3:
4944                            <1> ;     mov    [u.r0], ebx ; File/Device index/handle/descriptor
```

```
                        <1> ;       ; eax = device number (entry offset)
                        <1> ;       mov     ch, [eax+DEV_ACCESS] ; bit 0 = accessable by users
                        <1> ;                                    ; bit 1 = read access perm
                        <1> ;                                    ; bit 2 = write access perm
                        <1> ;                                    ; bit 3 = IOCTL permit to users
                        <1> ;                                    ; bit 4 = block device if set
                        <1> ;                                    ; bit 5 = 16 bit or 1024 byte
                        <1> ;                                    ; bit 6 = 32 bit or 2048 byte
                        <1> ;                                    ; bit 7 = installable device drv
                        <1> ;       test    ch, 1 ; accessable by normal users (except root)
                        <1> ;       jnz     short sysopen_dev_4 ; yes, permission has been given
                        <1> ;       cmp     byte [u.uid], 0 ; root?
                        <1> ;       jna     short sysopen_dev_4 ; superuser can open all devices
                        <1> ;sysopen_dev_perm_err:
                        <1> ;       mov     eax, ERR_DEV_ACCESS  ; 11 = 'permission denied !'
                        <1> ;       jmp     short sysopen_dev_err
                        <1> ;sysopen_dev_4:
                        <1> ;       shr     ch, 1 ; result: 1 = read, 2 = write, 3 = r & w
                        <1> ;       dec     cl  ; result: 1 = read, 2 = write
                        <1> ;       test    cl, ch
                        <1> ;       jz      short sysopen_dev_perm_err
                        <1> ;
                        <1> ;       shl     ch, 1 ; bit 0 = 0
                        <1> ;       ; eax = device number (entry offset)
                        <1> ;       call    device_open
                        <1> ;       jc      short sysopen_dev_perm_err
                        <1> ;
                        <1> ;       ; eax = device number (entry offset)
                        <1> ;       or      al, 80h ; set device bit (set bit 7 to 1)
                        <1> ;       mov     ebx, [u.r0]
                        <1> ;       mov     [ebx+u.fp], al      ; bit 7 (=1) points to device
                        <1> ;
                        <1> ;       jmp     sysret
                        <1>
                        <1> sysmkdir: ; < make directory >
                        <1>       ; 15/10/2016
                        <1>       ; 10/10/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>       ;          -derived from INT_21H.ASM-
                        <1>       ;            ("loc_INT21h_create_file")
                        <1>       ;     10/07/2011 (12/03/2011)
                        <1>       ;     INT 21h Function AH = 3Ch
                        <1>       ;     Create File
                        <1>       ;     INPUT
                        <1>       ;        CX = Attributes
                        <1>       ;          DS:DX= Address of zero terminaned path name
                        <1>       ;
                        <1>       ;
                        <1>       ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
                        <1>       ; 27/05/2013 - 02/08/2013 (Retro UNIX 8086 v1)
                        <1>       ;
                        <1>       ; 'sysmkdir' creates an empty directory whose name is
                        <1>       ; pointed to by arg 1. The mode of the directory is arg 2.
                        <1>       ; The special entries '.' and '..' are not present.
                        <1>       ; Errors are indicated if the directory already exists or
                        <1>       ; user is not the super user.
                        <1>       ;
                        <1>       ; Calling sequence:
                        <1>       ;     sysmkdir; name; mode
                        <1>       ; Arguments:
                        <1>       ;     name - points to the name of the directory
                        <1>       ;     mode - mode of the directory
                        <1>       ; Inputs: (arguments)
                        <1>       ; Outputs: -
                        <1>       ;    (sets 'directory' flag to 1;
                        <1>       ;    'set user id on execution' and 'executable' flags to 0)
                        <1>       ; ..........................................................
                        <1>       ;
                        <1>       ; Retro UNIX 8086 v1 modification:
                        <1>       ;       'sysmkdir' system call has two arguments; so,
                        <1>       ;     * 1st argument, name is pointed to by BX register
                        <1>       ;     * 2nd argument, mode is in CX register
                        <1>       ;
                        <1>       ; TRDOS 386 (10/10/2016)
                        <1>       ;
                        <1>       ; INPUT ->
                        <1>       ;        CL = Directory Attributes
                        <1>       ;           bit 0 (1) - Read only file/dir (R)
                        <1>       ;           bit 1 (1) - Hidden file/dir (H)
                        <1>       ;           bit 2 (1) - System file/dir (R)
                        <1>       ;           bit 3 (1) - Volume label/name (V)
                        <1>       ;           bit 4 (1) - Subdirectory (D)
                        <1>       ;           bit 5 (1) - File/Dir has been archived (A)
                        <1>       ;        CX = 0 -> create normal directory
                        <1>       ;          EBX = Pointer to directory name (ASCIIZ) -path-
                        <1>       ;
                        <1>       ; OUTPUT ->
                        <1>       ;          eax = First cluster of the new directory
                        <1>       ;          cf = 1 -> Error code in AL
                        <1>       ;
                        <1>       ; Modified Registers: EAX (at the return of system call)
                        <1>       ;
                        <1>       ; Note: If the file or directory is existing
                        <1>       ;     an access error will be returned.
                        <1>
0000DA63 6621C9         <1>       and     cx, cx ; if cx = 0 -> create a normal subdir
0000DA66 7413           <1>       jz      short sysmkdir_1
                        <1>
0000DA68 F6C110         <1>       test    cl, 10h ; if dir flags set, also use other flags
0000DA6B 0F85D3FDFFFF   <1>       jnz     sysmkdir_0 ; jump to head of 'syscreat'
                        <1>
                        <1>       ; CX has wrong flags
0000DA71 B817000000     <1>       mov     eax, ERR_INV_FLAGS
0000DA76 E9A4FEFFFF     <1>       jmp     sysopen_dev_err
                        <1>
                        <1> sysmkdir_1:
```

```
0000DA7B B110                <1>       mov    cl, 10h ; set subdir flag and reset other flags
0000DA7D E9C2FDFFFF          <1>       jmp    sysmkdir_0 ; jump to head of 'syscreat'
                             <1> sysmkdir_2:
                             <1>       ; jump from 'syscreat' ; from 'syscreat_1'
                             <1>       ;  CL = Directory attributes/flags
0000DA82 BE[508A0100]        <1>       mov    esi, FindFile_Name
0000DA87 E899D7FFFF          <1>       call   make_sub_directory
0000DA8C 0F82B0FEFFFF        <1>       jc     sysopen_err       ; NOTE: Old type (TRDOS 8086)
                             <1>                                ; error codes must be modified
                             <1>                                ; for next TRDOS 386 versions
                             <1>                                ; (10/10/2016)
                             <1>                                ; Old (MSDOS type)
                             <1>                                ; error codes (2011):
                             <1>                                ;  2 = file not found
                             <1>                                ;  3 = directory not found
                             <1>                                ;  5 = access denied
                             <1>                                ; 12 = no more files
                             <1>                                  ; 19 = disk write protected
                             <1>                                ; 39 = insufficient disk space
                             <1>                                ; 'sysdefs.s' ; 10/10/2016
                             <1>
0000DA92 A3[64030300]        <1>       mov    [u.r0], eax ; New sub dir's first cluster
                             <1>
0000DA97 E8F74F0000          <1>       call   reset_working_path
                             <1>
0000DA9C E92FF9FFFF          <1>       jmp    sysret
                             <1>
                             <1> sysclose: ;<close file>
                             <1>       ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
                             <1>       ;
                             <1>       ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
                             <1>       ; 22/05/2013 - 26/05/2013 (Retro UNIX 8086 v1)
                             <1>       ;
                             <1>       ; 'sysclose', given a file descriptor in 'u.r0', closes the
                             <1>       ; associated file. The file descriptor (index to 'u.fp' list)
                             <1>       ; is put in r1 and 'fclose' is called.
                             <1>       ;
                             <1>       ; Calling sequence:
                             <1>       ;     sysclose
                             <1>       ; Arguments:
                             <1>       ;     -
                             <1>       ; Inputs: *u.r0 - file descriptor
                             <1>       ; Outputs: -
                             <1>       ; ..........................................................
                             <1>       ;
                             <1>       ; Retro UNIX 8086 v1 modification:
                             <1>       ;      The user/application program puts file descriptor
                             <1>       ;       in BX register as 'sysclose' system call argument.
                             <1>       ;      (argument transfer method 1)
                             <1>
                             <1>       ; TRDOS 386 (06/10/2016)
                             <1>       ;
                             <1>         ; INPUT ->
                             <1>         ;      EBX = File Handle/Number (file index) (AL)
                             <1>       ; OUTPUT ->
                             <1>       ;          cf = 0 -> EAX = 0
                             <1>       ;          cf = 1 -> Error code in EAX (ERR_FILE_NOT_OPEN)
                             <1>       ;
                             <1>       ; Modified Registers: EAX (at the return of system call)
                             <1>       ;
                             <1>
0000DAA1 89D8                <1>       mov    eax, ebx
0000DAA3 31DB                <1>       xor    ebx, ebx
0000DAA5 891D[64030300]      <1>       mov    [u.r0], ebx ; 0  ; return value of EAX
0000DAAB E8822F0000          <1>       call   fclose
0000DAB0 0F831AF9FFFF        <1>       jnc    sysret
0000DAB6 B80A000000          <1>       mov    eax, ERR_FILE_NOT_OPEN ; file not open !
0000DABB A3[C8030300]        <1>       mov    [u.error], eax ;
0000DAC0 A3[64030300]        <1>       mov    [u.r0], eax ; ! invalid handle !
0000DAC5 E9E6F8FFFF          <1>       jmp    error
                             <1>
                             <1> sysread: ; < read from file >
                             <1>       ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
                             <1>       ;          -derived from INT_21H.ASM-
                             <1>       ;            ("loc_INT21h_read_file")
                             <1>       ;     13/03/2011 (05/03/2011)
                             <1>       ;     INT 21h Function AH = 3Fh
                             <1>       ;     Read from a File
                             <1>       ;     INPUT
                             <1>       ;        BX = File Handle
                             <1>       ;        CX = Number of bytes to read
                             <1>       ;          DS:DX= Buffer address
                             <1>       ;
                             <1>       ; Note: TRDOS 386 'sysread' has been derived from
                             <1>       ;     Retro UNIX 386 v1 'sysread', except a few
                             <1>       ;     code modifications.
                             <1>       ;
                             <1>       ; 13/05/2015 (Retro UNIX 386 v1)
                             <1>       ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
                             <1>       ; 23/05/2013 (Retro UNIX 8086 v1)
                             <1>       ;
                             <1>       ; 'sysread' is given a buffer to read into and the number of
                             <1>       ; characters to be read. If finds the file from the file
                             <1>       ; descriptor located in *u.r0 (r0). This file descriptor
                             <1>       ; is returned from a successful open call (sysopen).
                             <1>       ; The i-number of file is obtained via 'rw1' and the data
                             <1>       ; is read into core via 'readi'.
                             <1>       ;
                             <1>       ; Calling sequence:
                             <1>       ;     sysread; buffer; nchars
                             <1>       ; Arguments:
                             <1>       ;     buffer - location of contiguous bytes where
                             <1>       ;              input will be placed.
                             <1>       ;     nchars - number of bytes or characters to be read.
                             <1>       ; Inputs: *u.r0 - file descriptor (& arguments)
```

```
                                <1>       ; Outputs: *u.r0 - number of bytes read.
                                <1>       ; ..........................................................
                                <1>       ;
                                <1>       ; Retro UNIX 8086 v1 modification:
                                <1>       ;       'sysread' system call has three arguments; so,
                                <1>       ;     * 1st argument, file descriptor is in BX register
                                <1>       ;     * 2nd argument, buffer address/offset in CX register
                                <1>       ;     * 3rd argument, number of bytes is in DX register
                                <1>       ;
                                <1>       ;     AX register (will be restored via 'u.r0') will return
                                <1>       ;     to the user with number of bytes read.
                                <1>       ;
                                <1>       ; TRDOS 386 (05/10/2016)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;         EBX = File handle (descriptor/index)
                                <1>       ;         ECX = Buffer address
                                <1>       ;           EDX = Number of bytes
                                <1>       ; OUTPUT ->
                                <1>       ;           EAX = Number of bytes have been read
                                <1>       ;           cf = 1 -> Error code in AL
                                <1>       ;
                                <1>       ; Modified Registers: EAX (at the return of system call)
                                <1>       ;
                                <1>
                                <1>       ; EBX = File descriptor
0000DACA E8B32F0000             <1>       call   getf1
0000DACF 7273                   <1>       jc     short device_read ; read data from device
                                <1>       ; EAX = First cluster of the file
                                <1>
0000DAD1 E83F000000             <1>       call   rw1
0000DAD6 730A                   <1>       jnc    short sysread_0
                                <1>
0000DAD8 A3[64030300]           <1>       mov    [u.r0], eax ; error code
0000DADD E9CEF8FFFF             <1>       jmp    error
                                <1>
                                <1> sysread_0:
0000DAE2 E891350000             <1>       call   readi
0000DAE7 EB1D                   <1>       jmp    short rw0
                                <1>
                                <1> syswrite: ; < write to file >
                                <1>       ; 23/10/2016
                                <1>       ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ;           -derived from INT_21H.ASM-
                                <1>       ;             ("loc_INT21h_write_file")
                                <1>       ;     13/03/2011 (05/03/2011)
                                <1>       ;     INT 21h Function AH = 40h
                                <1>       ;     Write to a File
                                <1>       ;     INPUT
                                <1>       ;        BX = File Handle
                                <1>       ;        CX = Number of bytes to write
                                <1>       ;          DS:DX= Buffer address
                                <1>       ;
                                <1>       ; Note: TRDOS 386 'sysrwrite' has been derived from
                                <1>       ;     Retro UNIX 386 v1 'syswrite', except a few
                                <1>       ;     code modifications.
                                <1>       ;
                                <1>
                                <1>       ; 13/05/2015 (Retro UNIX 386 v1)
                                <1>       ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
                                <1>       ; 23/05/2013 (Retro UNIX 8086 v1)
                                <1>       ;
                                <1>       ; 'syswrite' is given a buffer to write onto an output file
                                <1>       ; and the number of characters to write. If finds the file
                                <1>       ; from the file descriptor located in *u.r0 (r0). This file
                                <1>       ; descriptor is returned from a successful open or create call
                                <1>       ; (sysopen or syscreat). The i-number of file is obtained via
                                <1>       ; 'rw1' and buffer is written on the output file via 'write'.
                                <1>       ;
                                <1>       ; Calling sequence:
                                <1>       ;     syswrite; buffer; nchars
                                <1>       ; Arguments:
                                <1>       ;     buffer - location of contiguous bytes to be writtten.
                                <1>       ;     nchars - number of characters to be written.
                                <1>       ; Inputs: *u.r0 - file descriptor (& arguments)
                                <1>       ; Outputs: *u.r0 - number of bytes written.
                                <1>       ; ..........................................................
                                <1>       ;
                                <1>       ; Retro UNIX 8086 v1 modification:
                                <1>       ;       'syswrite' system call has three arguments; so,
                                <1>       ;     * 1st argument, file descriptor is in BX register
                                <1>       ;     * 2nd argument, buffer address/offset in CX register
                                <1>       ;     * 3rd argument, number of bytes is in DX register
                                <1>       ;
                                <1>       ;     AX register (will be restored via 'u.r0') will return
                                <1>       ;     to the user with number of bytes written.
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;         EBX = File handle (descriptor/index)
                                <1>       ;         ECX = Buffer address
                                <1>       ;           EDX = Number of bytes
                                <1>       ; OUTPUT ->
                                <1>       ;           EAX = Number of bytes have been written
                                <1>       ;           cf = 1 -> Error code in AL
                                <1>       ;
                                <1>       ; Modified Registers: EAX (at the return of system call)
                                <1>       ;
                                <1>
                                <1>       ; EBX = File descriptor
0000DAE9 E8942F0000             <1>       call   getf1
0000DAEE 7254                   <1>       jc     short device_write ; write data to device
                                <1>       ; EAX = First cluster of the file
                                <1>       ; EBX = File number  (Open file number) ; 23/10/2016
                                <1>
0000DAF0 E820000000             <1>       call   rw1
```

```
5260 0000DAF5 730A                <1>       jnc    short syswrite_0
5261 0000DAF7 A3[64030300]        <1>       mov    [u.r0], eax ; error code
5262 0000DAFC E9AFF8FFFF          <1>       jmp    error
5263                              <1>
5264                              <1> syswrite_0:
5265 0000DB01 E89E3C0000          <1>       call   writei
5266                              <1> rw0: ; 1:
5267 0000DB06 A1[8C030300]        <1>       mov    eax, [u.nread]
5268 0000DB0B A3[64030300]        <1>       mov    [u.r0], eax
5269 0000DB10 E9BBF8FFFF          <1>       jmp    sysret
5270                              <1> rw1:
5271                              <1>       ; 17/04/2021 (TRDOS 386 v2.0.4)
5272                              <1>       ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
5273                              <1>       ; 14/05/2015 (Retro UNIX 386 v1)
5274                              <1>       ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
5275                              <1>       ; 23/05/2013 - 24/05/2013 (Retro UNIX 8086 v1)
5276                              <1>       ; System call registers: ebx, ecx, edx (through 'sysenter')
5277                              <1>       ;
5278                              <1>       ; EBX = File descriptor
5279                              <1>       ;call  getf1 ; calling point in 'getf' from 'rw1'
5280                              <1>       ;jc    short device_rw ; read/write data from/to device
5281                              <1>       ; EAX = First cluster of the file
5282                              <1>
5283 0000DB15 83F802              <1>       cmp    eax, 2
5284 0000DB18 7217                <1>       jb     short rw2
5285                              <1>       ;
5286 0000DB1A 890D[84030300]      <1>       mov    [u.base], ecx       ; buffer address/offset
5287                              <1>                                  ;(in the user's virtual memory space)
5288 0000DB20 8915[88030300]      <1>       mov    [u.count], edx
5289                              <1>
5290 0000DB26 C705[C8030300]0000- <1>       mov    dword [u.error], 0 ; reset the last error code
5290 0000DB2E 0000                <1>
5291 0000DB30 C3                  <1>       retn
5292                              <1> rw2:
5293 0000DB31 B80A000000          <1>       mov    eax, ERR_FILE_NOT_OPEN ; file not open !
5294                              <1>       ;mov   dword [u.error], eax
5295                              <1>       ;retn
5296                              <1>       ; 17/04/2021
5297 0000DB36 EB06                <1>       jmp    short rw4
5298                              <1> rw3:
5299 0000DB38 B80B000000          <1>       mov    eax, ERR_FILE_ACCESS ; permission denied !
5300 0000DB3D F9                  <1>       stc
5301                              <1> rw4:  ; 17/04/2021
5302 0000DB3E A3[C8030300]        <1>       mov    dword [u.error], eax
5303 0000DB43 C3                  <1>       retn
5304                              <1>
5305                              <1>       ; 17/04/2021 (temporary)
5306                              <1> device_write:
5307                              <1> device_read:
5308                              <1>       ; 17/04/2021 - TRDOS 386 v2.0.4
5309                              <1>       ;      (temporary modifications)
5310                              <1>       ;
5311                              <1>       ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
5312                              <1>       ; cl = DEV_OPENMODE ; open mode
5313                              <1>       ; ch = DEV_ACCESS   ; access flags
5314                              <1>       ; al = DEV_DRIVER   ; device number (eax)
5315                              <1>
5316                              <1>       ; 17/04/2021 (temporary)
5317 0000DB44 EBEB                <1>       jmp    short rw2 ; file not open
5318                              <1>
5319                              <1> ;     test   cl, 1 ; 1 = read, 2 = write, 3 = read&write
5320                              <1> ;     jz     short rw3
5321                              <1> ;
5322                              <1> ;     mov    ebx, eax
5323                              <1> ;     shl    bx, 2 ; *4
5324                              <1> ;
5325                              <1> ;     test   ch, 80h ; bit 7, installable device driver flag
5326                              <1> ;     jz     short d_read_2 ; Kernel device
5327                              <1> ;     ; installable device
5328                              <1> ;d_read_1:
5329                              <1> ;       jmp  dword [ebx+IDEV_RADDR-4]
5330                              <1> ;d_read_2:
5331                              <1> ;     jmp    dword [ebx+KDEV_RADDR-4]
5332                              <1>
5333                              <1> ;device_write:
5334                              <1>       ; 17/04/2021 - TRDOS 386 v2.0.4
5335                              <1>       ;      (temporary modifications)
5336                              <1>       ;
5337                              <1>       ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
5338                              <1>       ; cl = DEV_OPENMODE ; open mode
5339                              <1>       ; ch = DEV_ACCESS   ; access flags
5340                              <1>       ; al = DEV_DRIVER   ; device number (eax)
5341                              <1>
5342                              <1>       ; 17/04/2021 (temporary)
5343                              <1>       ;jmp   short rw2 ; file not open
5344                              <1>
5345                              <1> ;     test   cl, 2 ; 1 = read, 2 = write, 3 = read&write
5346                              <1> ;     jz     short rw3
5347                              <1> ;
5348                              <1> ;     mov    ebx, eax
5349                              <1> ;     shl    bx, 2 ; *4
5350                              <1> ;
5351                              <1> ;     test   ch, 80h ; bit 7, installable device driver flag
5352                              <1> ;     jz     short d_write_2 ; Kernel device
5353                              <1> ;     ; installable device
5354                              <1> ;d_write_1:
5355                              <1> ;       jmp  dword [ebx+IDEV_WADDR-4]
5356                              <1> ;d_write_2:
5357                              <1> ;     jmp    dword [ebx+KDEV_WADDR-4]
5358                              <1>
5359                              <1> sysemt: ; enable (or disable) multi tasking -time sharing-
5360                              <1>       ;
5361                              <1>       ; 23/05/2016 - TRDOS 386 (TRDOS v2.0)
5362                              <1>       ; 14/05/2015 (Retro UNIX 386 v1)
5363                              <1>       ; 10/12/2013 - 20/04/2014 (Retro UNIX 8086 v1)
```

```
                                        <1>       ;
                                        <1>       ; Retro UNIX 8086 v1 modification:
                                        <1>       ;     'Enable Multi Tasking'  system call instead
                                        <1>       ;     of 'Emulator Trap' in original UNIX v1 for PDP-11.
                                        <1>       ;
                                        <1>       ; Retro UNIX 8086 v1 feature only!
                                        <1>       ;     Using purpose: Kernel will start without time-out
                                        <1>       ;     (internal clock/timer) functionality.
                                        <1>       ;     Then etc/init will enable clock/timer for
                                        <1>       ;     multi tasking.
                                        <1>       ;
                                        <1>       ; INPUT ->
                                        <1>       ;     BL = 0 -> disable multi tasking
                                        <1>       ;     BL > 1 -> enable multi tasking (time sharing)
                                        <1>       ; OUTPUT ->
                                        <1>       ;     none
                                        <1>       ;
                                        <1>       ;  Note: Multi tasking is disabled during system
                                        <1>       ;       initialization, it must be enabled by using
                                        <1>       ;       this system call. (Otherwise, running proces
                                        <1>       ;       will not be changed by another process within
                                        <1>       ;       run time sequence/schedule, if running process
                                        <1>       ;       will not 'release' itself. Only 'wakeup' procedure
                                        <1>       ;       for waiting processes and programmed timer events
                                        <1>       ;       for other processes can change running process
                                        <1>       ;       while multi tasking is disabled.) ** 23/05/2016 **
                                        <1>
0000DB46 803D[B0030300]00               <1>       cmp    byte [u.uid], 0 ; root ?
                                        <1>       ;ja    error
0000DB4D 0F87A2F9FFFF                   <1>       ja     badsys ; 14/05/2015
                                        <1>       ;
0000DB53 FA                             <1>       cli
0000DB54 881D[3A8D0100]                 <1>       mov    [multi_tasking], bl ; 0 to disable, >0 to enable
0000DB5A E971F8FFFF                     <1>       jmp    sysret
                                        <1>
                                        <1> systimer:
                                        <1>       ; 02/01/2017
                                        <1>       ; 21/12/2016
                                        <1>       ; 19/12/2016
                                        <1>       ; 10/12/2016 (callback)
                                        <1>       ; 10/06/2016
                                        <1>       ; 07/06/2016
                                        <1>       ; 06/06/2016
                                        <1>       ; 21/05/2016
                                        <1>       ; 19/05/2016
                                        <1>       ; 18/05/2016 - TRDOS 386 (TRDOS v2.0)
                                        <1>       ; (TRDOS 386 feature only!)
                                        <1>       ;
                                        <1>       ; (start or stop timer event(s))
                                        <1>       ;
                                        <1>       ; INPUT ->
                                        <1>       ;     BL = Signal return byte (response byte)
                                        <1>       ;          (Any requested value between 0 and 255)
                                        <1>       ;          (Kernel will put it at the requested address)
                                        <1>       ;     BH = Time count unit
                                        <1>       ;          0 = Stop timer event
                                        <1>       ;          1 = 18.2 ticks per second
                                        <1>       ;          2 = 10 milliseconds
                                        <1>       ;          3 = 1 second (for real time clock interrupt)
                                        <1>       ;          4 = time/tick count in current time count unit
                                        <1>       ;         // 10/12/2016
                                        <1>       ;         80h = Stop timer event (callback method)
                                        <1>       ;         81h = 18.2 ticks per second, callback method
                                        <1>       ;         82h = 10 milliseconds, callback method
                                        <1>       ;         83h = 1 second (for RTC int), callback method
                                        <1>       ;         84h = current time count unit, callback method
                                        <1>       ;
                                        <1>       ;         Note: Only 03h or 83h will set real time clock
                                        <1>       ;              (RTC) events (Others are for PIT events)!
                                        <1>       ;
                                        <1>       ;     NOTE: If callback (user service) method is used,
                                        <1>       ;         EDX will point to the return address (of service
                                        <1>       ;         procedure) in user's space instead of signal
                                        <1>       ;         response byte address. (TRDOS 386 kernel will
                                        <1>       ;         direct the cpu to that address -in user's space-
                                        <1>       ;         at the return of system call or interrupt
                                        <1>       ;         just after the adjusted count/time is elapsed.)
                                        <1>       ;         User's sevice routine must be ended with a
                                        <1>       ;         'iret'. Normal return addresses from system
                                        <1>       ;         calls or and interrupts will be kept same except
                                        <1>       ;         the timer returns.
                                        <1>       ;
                                        <1>       ;     BH = 0 -> Stop timer event
                                        <1>       ;     BL = Timer event number (1 to 255) if BH = 0
                                        <1>       ;          If BL = 0, all timer events (which are belongs
                                        <1>       ;           to running process) will be stopped
                                        <1>       ;     ECX = Time/Tick count (depending on time count unit)
                                        <1>       ;     EDX = Signal return (Response) byte address
                                        <1>       ;           (virtual address in user's memory space)
                                        <1>       ; OUTPUT ->
                                        <1>       ;     AL = Timer event number   (1 to 255) (max. value = 16)
                                        <1>       ;     IF BH Input = 0 & CF = 0 & AL = 0 ->
                                        <1>       ;          timer event(s) has/have been stopped/finished
                                        <1>       ;     CF = 1 & AL = 0 -> no timer setting space to set
                                        <1>       ;     CF = 1 & AL > 0 -> timer count unit is not usable
                                        <1>       ;
                                        <1>       ;     NOTE: To modify a time count for a user function,
                                        <1>       ;           at first, current timer event must be stopped
                                        <1>       ;           then a new timer event (which is related with
                                        <1>       ;           same user function) must be started.
                                        <1>       ;
                                        <1>       ;           Signal return (response) byte may be used for
                                        <1>       ;           several purposes. Kernel will put this value
                                        <1>       ;           to requested address during timer interrupt,
                                        <1>       ;           program/user can check this value to understand
```

```
                                        <1>       ;           which event has been occurred and what is changed.
                                        <1>       ;           (Multi timer events can share same signal address)
                                        <1>       ;
                                        <1>       ;     NOTE: If the process is running while the time count
                                        <1>       ;           is reached, kernel will put signal return (response)
                                        <1>       ;           byte value at requested address during timer
                                        <1>       ;           interrupt and the process will continue to run.
                                        <1>       ;           Program/process must call (jump to) it's timer event
                                        <1>       ;           function as required, for checking the timer event
                                        <1>       ;           status via signal return (response) byte address.
                                        <1>       ;
                                        <1>       ;           If the process is not running (waiting or sleeping
                                        <1>       ;           or released) while the time count is reached,
                                        <1>       ;           it is restarted from where it left, to ensure
                                        <1>       ;           proper multi media (video, audio, clock, timer)
                                        <1>       ;           functionality.
                                        <1>       ;
                                        <1>       ;           (It is better to use 'syswait' or 'syssleep',
                                        <1>       ;           or 'sysrele' system call just after the timer
                                        <1>       ;           function. Otherwise, timer events may block other
                                        <1>       ;           processes which are not using timer events.)
                                        <1>       ;
                                        <1>       ; Timer Event Structure: (max. 16 timer events, 16*16 bytes)
                                        <1>       ;      Owner:              resb 1 ; 0 = free
                                        <1>       ;                         ;>0 = process number (u.uno)
                                        <1>       ;     Calback:     resb 1 ; 1 = callback, 0 = response byte
                                        <1>       ;     Interrupt:      resb 1 ; 0 = Timer interrupt (or none)
                                        <1>       ;                         ; 1 = Real Time Clock interrupt
                                        <1>       ;     Response:       resb 1 ; 0 to 255, signal return value
                                        <1>       ;     Count Limit: resd 1 ; count of ticks (total/set)
                                        <1>       ;     Current Count:      resd 1 ; count of ticks (current)
                                        <1>       ;     Response Addr:  resd 1 ; response byte (pointer) address
                                        <1>       ;
                                        <1>
                                        <1>       ; 19/12/2016 (timer callback)
0000DB5F C605[448F0100]00               <1>       mov   byte [tcallback], 0
0000DB66 C605[458F0100]00               <1>       mov   byte [trtc], 0
0000DB6D C705[D0030300]0000-            <1>       mov   dword [u.tcb], 0 ; this is not necessary...
0000DB75 0000                           <1>
                                        <1>
0000DB77 80FF80                         <1>       cmp   bh, 80h
0000DB7A 7225                           <1>       jb    short systimer_cb2
0000DB7C 7704                           <1>       ja    short systimer_cb0
                                        <1>
0000DB7E 31D2                           <1>       xor   edx, edx ; 0, reset callback address
0000DB80 EB0B                           <1>       jmp   short systimer_cb1
                                        <1>
                                        <1> systimer_cb0:
0000DB82 80FF84                         <1>       cmp   bh, 84h
0000DB85 7764                           <1>       ja    short systimer_5 ;  undefined, error
                                        <1>
                                        <1>       ;mov  byte [tcallback], 1 ; 19/12/2016
0000DB87 FE05[448F0100]                 <1>       inc   byte [tcallback]
                                        <1>
                                        <1> systimer_cb1:
0000DB8D 0FB635[B3030300]               <1>       movzx esi, byte [u.uno] ; process number
0000DB94 66C1E602                       <1>       shl   si, 2
0000DB98 8996[0C010300]                 <1>       mov   [esi+p.tcb-4], edx ; set process timer callback address
                                        <1>                                ; (overwrite prev value if it is set!)
0000DB9E 80E77F                         <1>       and   bh, 7Fh
                                        <1>
                                        <1> systimer_cb2:
0000DBA1 80FF02                         <1>       cmp   bh, 2
0000DBA4 7445                           <1>       je    short systimer_5  ; only 18.2 ticks per second is usable
                                        <1>                               ; 10 milliseconds (100 Hertz) timer
                                        <1>                               ; will be set later (18/05/2016)
0000DBA6 774B                           <1>       ja    short systimer_6
                                        <1>
0000DBA8 20FF                           <1>       and   bh, bh
0000DBAA 0F84BA000000                   <1>       jz    systimer_9        ; stop timer event(s)
                                        <1>
                                        <1>       ; bh = 1 (timer interrupt, 18.2 Hz, IBM PC/AT ROMBIOS default)
                                        <1>
                                        <1> systimer_19:
0000DBB0 B00A                           <1>       mov   al, 10 ; (*)
                                        <1>
                                        <1> systimer_0:
0000DBB2 B710                           <1>       mov   bh, 16
                                        <1>       ;
0000DBB4 383D[3B8D0100]                 <1>       cmp   [timer_events], bh ; 16 ; 07/06/2016
0000DBBA 7319                           <1>       jnb   short systimer_3  ; max. 16 timer events
                                        <1>       ;
0000DBBC 50                             <1>       push  eax ; (*)
                                        <1>
0000DBBD BF[60040300]                   <1>       mov   edi, timer_set  ; beginning address of timer events
                                        <1>                             ; setting space
0000DBC2 30C0                           <1>       xor   al, al ; 0
                                        <1> systimer_1:
0000DBC4 FEC0                           <1>       inc   al
0000DBC6 803F00                         <1>       cmp   byte [edi], 0       ; is it free space ?
0000DBC9 7639                           <1>       jna   short systimer_7 ; yes
0000DBCB FECF                           <1>       dec   bh
0000DBCD 7405                           <1>       jz    short systimer_2
0000DBCF 83C710                         <1>       add   edi, 16
0000DBD2 EBF0                           <1>       jmp   short  systimer_1 ; next event space
                                        <1>
                                        <1> systimer_2:
0000DBD4 58                             <1>       pop   eax ; (*) discard
                                        <1> systimer_3:
0000DBD5 C605[64030300]00               <1>       mov   byte [u.r0], 0
                                        <1> systimer_4:
0000DBDC C705[C8030300]1B00-            <1>       mov   dword [u.error], ERR_MISC
0000DBE4 0000                           <1>
                                        <1>                                ; one of miscellaneous/other errors
0000DBE6 E9C5F7FFFF                     <1>       jmp   error ; cf -> 1
```

```
                                <1>
                                <1> systimer_5:
0000DBEB 883D[64030300]         <1>       mov    [u.r0], bh ; Time count unit (=2 or >3)
0000DBF1 EBE9                   <1>       jmp    short systimer_4 ; 07/06/2016
                                <1>
                                <1> systimer_6:
0000DBF3 80FF04                 <1>       cmp    bh, 4
0000DBF6 77F3                   <1>       ja     short systimer_5  ; undefined time count unit
                                <1>       ;jb    short systimer_16
                                <1>
                                <1>       ;mov   al, 1  ; default (use current timer unit)
                                <1>                     ; countdown value is in ECX !
                                <1>                     ; max. value of ecx = 4294967296/10
                                <1>       ;jmp  short systimer_0
                                <1>       ;jmp   short systimer_19
0000DBF8 74B6                   <1>       je     short systimer_19
                                <1>
                                <1> systimer_16:
                                <1>       ; bh = 3
                                <1>       ; timer event via real time clock interrupt
                                <1>       ; interrupt/update frequency: 1 Hz (1 tick per second)
                                <1>
0000DBFA B0B6                   <1>       mov    al, 182 ; (*) ; 18.2 * 10
0000DBFC FE05[458F0100]         <1>       inc    byte [trtc] ; timer event via real time clock
0000DC02 EBAE                   <1>       jmp    short systimer_0
                                <1>
                                <1> systimer_7:
0000DC04 A2[64030300]           <1>       mov    [u.r0], al ; timer event number
                                <1>       ;
                                <1>       ; edi = address of empty timer event area
0000DC09 A0[B3030300]           <1>       mov    al, [u.uno]
0000DC0E FA                     <1>       cli    ; disable interrupts
0000DC0F AA                     <1>       stosb  ; process number
0000DC10 A0[448F0100]           <1>       mov    al, [tcallback] ; timer callback flag
0000DC15 AA                     <1>       stosb  ; 1= callback method, 0= signal response byte method
0000DC16 A0[458F0100]           <1>       mov    al, [trtc] ; timer interrupt type
0000DC1B AA                     <1>       stosb  ; 1= real time clock, 0= programmable interval timer
0000DC1C 88D8                   <1>       mov    al, bl ; Signal return (Response) value
0000DC1E AA                     <1>       stosb   ; response byte
0000DC1F 58                     <1>       pop    eax ; (*) ; 10 or 182
0000DC20 89D3                   <1>       mov    ebx, edx ; virtual address for response/signal byte
0000DC22 F7E1                   <1>       mul    ecx
                                <1>       ; (eax = 10 * count of 18.2 Hz timer ticks)
                                <1>       ; (count down step = 10)
0000DC24 AB                     <1>       stosd  ; count limit (reset value)
0000DC25 AB                     <1>       stosd  ; current count value
                                <1>
                                <1>       ; 19/12/2016
0000DC26 803D[448F0100]00       <1>       cmp    byte [tcallback], 0 ; timer callback method ?
0000DC2D 7604                   <1>       jna    short systimer_17 ; no
0000DC2F 89D8                   <1>       mov    eax, ebx ; virtual address for callback routine
0000DC31 EB0D                   <1>       jmp    short systimer_18
                                <1>
                                <1> systimer_17: ; signal response byte method
                                <1>       ; ebx = virtual address
                                <1>       ; [u.pgdir] = page directory's physical address
                                <1>       ; 20/02/2017
0000DC33 FE05[468F0100]         <1>       inc     byte [no_page_swap] ; 1
                                <1>                     ; Do not add this page to swap queue
                                <1>                     ; and remove it from swap queue if it is
                                <1>                     ; on the queue.
0000DC39 E82482FFFF             <1>       call   get_physical_addr
0000DC3E 721A                   <1>       jc     short systimer_8 ; 07/06/2016
                                <1>       ; eax = physical address of the virtual address in user's space
                                <1> systimer_18:
0000DC40 AB                     <1>       stosd  ; response addr (physical) or callback addr (virtual)
0000DC41 FE05[3B8D0100]         <1>       inc    byte [timer_events] ; 07/06/201
                                <1>       ; 02/01/2017
0000DC47 0FB605[B3030300]       <1>       movzx  eax, byte [u.uno]
0000DC4E FE80[FF000300]         <1>       inc    byte [eax+p.timer-1]
                                <1>       ;
0000DC54 FB                     <1>       sti    ; enable interrupts
0000DC55 E976F7FFFF             <1>       jmp    sysret
                                <1>
                                <1> systimer_8:
                                <1>       ; 10/06/2016
                                <1>       ; 07/06/2016
0000DC5A 28C0                   <1>       sub    al, al ; 0
0000DC5C 8847F4                 <1>       mov    [edi-12], al ; clear process number (free timer event)
                                <1>       ;mov   dword [edi], eax ; 0
0000DC5F FB                     <1>       sti
0000DC60 A2[64030300]           <1>       mov    [u.r0], al ; 0
0000DC65 E946F7FFFF             <1>       jmp    error
                                <1>
                                <1> systimer_9:
                                <1>       ; 10/06/2016
                                <1>       ; 07/06/2016
0000DC6A 28C0                   <1>       sub    al, al
0000DC6C A2[64030300]           <1>       mov    byte [u.r0], al ; 0
0000DC71 3805[3B8D0100]         <1>       cmp    byte [timer_events], al ;  0
0000DC77 7631                   <1>       jna    short systimer_12
                                <1>
                                <1>       ; Note: ecx and edx are undefined here
                                <1>       ;      (for stop timer function)
                                <1>
0000DC79 BE[60040300]           <1>       mov    esi, timer_set  ; beginning address of timer events
                                <1>                            ; setting space
0000DC7E A0[B3030300]           <1>       mov    al, [u.uno]
                                <1>
0000DC83 B710                   <1>       mov    bh, 16
                                <1>
0000DC85 08DB                   <1>       or     bl, bl
0000DC87 7544                   <1>       jnz    short systimer_15
                                <1>
                                <1>       ; clear timer event areas belong to current process
```

```
                                <1>       ; (for stopping all timer events belong to current process)
0000DC89 FA                     <1>       cli    ; disable interrupts
                                <1> systimer_10:
                                <1>       ; 10/06/2016
                                <1>       ; 07/06/2016
0000DC8A 8A26                   <1>       mov    ah, [esi]
0000DC8C 08E4                   <1>       or     ah, ah ; 0 ?
0000DC8E 7411                   <1>       jz     short systimer_11
0000DC90 38C4                   <1>       cmp    ah, al ; is the process number (owner) same ?
0000DC92 750D                   <1>       jne   short systimer_11 ; no
                                <1>
                                <1>       ;mov   byte [esi], 0
0000DC94 66C7060000             <1>       mov    word [esi], 0 ; clear
                                <1>       ;mov   dword [esi+12], 0 ; clear
                                <1>
0000DC99 FE0D[3B8D0100]         <1>       dec    byte [timer_events]
0000DC9F 7409                   <1>       jz     short systimer_12
                                <1>
                                <1> systimer_11:
0000DCA1 FECF                   <1>       dec    bh
0000DCA3 7405                   <1>       jz     short systimer_12
0000DCA5 83C610                 <1>       add    esi, 16
0000DCA8 EBE0                   <1>       jmp    short systimer_10
                                <1>
                                <1> systimer_12:
0000DCAA 0FB635[B3030300]       <1>       movzx  esi, byte [u.uno]
0000DCB1 08DB                   <1>       or     bl, bl ; all timer events or one timer event ?
0000DCB3 740C                   <1>       jz     short systimer_13
0000DCB5 8A9E[FF000300]         <1>       mov    bl, [esi+p.timer-1]
0000DCBB 20DB                   <1>       and    bl, bl ; previous number of timer events for the process
0000DCBD 7408                   <1>       jz     short systimer_14
0000DCBF FECB                   <1>       dec    bl  ; previous number of timer events for the process - 1
                                <1> systimer_13:
0000DCC1 889E[FF000300]         <1>       mov    [esi+p.timer-1], bl ; 0 ; no timer events for process
                                <1> systimer_14:
0000DCC7 FB                     <1>       sti    ; enable interrupts
0000DCC8 E903F7FFFF             <1>       jmp    sysret
                                <1>
                                <1> systimer_15:
0000DCCD 38FB                   <1>       cmp    bl, bh ; 16
0000DCCF 0F8707FFFFFF           <1>       ja    systimer_4      ; max. 16 timer events !
                                <1>       ;
0000DCD5 88DA                   <1>       mov    dl, bl
0000DCD7 FECA                   <1>       dec    dl  ; 16 -> 15 ... 1 -> 0
0000DCD9 C0E204                 <1>       shl    dl, 4 ; * 16
0000DCDC 0FB6FA                 <1>       movzx  edi, dl
0000DCDF 01F7                   <1>       add    edi, esi ; timer_set
                                <1>
0000DCE1 3A07                   <1>       cmp    al, [edi] ; process number
0000DCE3 0F85F3FEFFFF           <1>       jne   systimer_4
                                <1>
                                <1>       ; same process ID
0000DCE9 FA                     <1>       cli    ; disable interrupts
                                <1>       ; 10/06/2016 ; 02/01/2017
                                <1>       ;mov   byte [edi], 0
0000DCEA 66C7070000             <1>       mov    word [edi], 0 ; clear
                                <1>       ;mov   dword [edi+12], 0 ; clear
0000DCEF FE0D[3B8D0100]         <1>       dec    byte [timer_events]
0000DCF5 EBB3                   <1>       jmp    short systimer_12
                                <1>
                                <1> sysvideo: ; VIDEO DATA TRANSFER FUNCTIONS
                                <1>       ; 06/03/2021
                                <1>       ; 02/03/2021
                                <1>       ; 28/02/2021
                                <1>       ; 27/02/2021
                                <1>       ; 26/02/2021
                                <1>       ; 25/02/2021
                                <1>       ; 21/02/2021, 22/02/2021, 23/02/2021
                                <1>       ; 15/02/2021, 16/02/2021, 18/02/2021
                                <1>       ; 10/02/2021, 11/02/2021, 12/02/2021
                                <1>       ; 07/02/2021, 08/02/2021
                                <1>       ; 01/02/2021, 02/02/2021, 05/02/2021
                                <1>       ; 29/01/2021, 30/01/2021, 31/01/2021
                                <1>       ; 23/01/2021, 24/01/2021, 28/01/2021
                                <1>       ; 18/01/2021, 19/01/2021, 22/01/2021
                                <1>       ; 04/01/2021, 10/01/2021, 11/01/2021
                                <1>       ; 01/01/2021, 02/01/2021, 03/01/2021
                                <1>       ; 28/12/2020, 29/12/2020, 30/12/2020
                                <1>       ; 25/12/2020, 26/12/2020
                                <1>       ; 21/12/2020, 23/12/2020
                                <1>       ; 12/12/2020, 14/12/2020
                                <1>       ; 10/12/2020, 11/12/2020
                                <1>       ; 03/12/2020, 04/12/2020
                                <1>       ; 22/11/2020, 23/11/2020
                                <1>       ; 21/11/2020 (TRDOS 386 v2.0.3)
                                <1>       ; 12/05/2017
                                <1>       ; 11/07/2016
                                <1>       ; 13/06/2016
                                <1>       ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ;
                                <1>       ; VIDEO DATA TRANSFER FUNCTIONS:
                                <1>       ;
                                <1>       ; Inputs:
                                <1>       ;            ; 07/02/2021
                                <1>       ;     BH = 0 = VIDEO BIOS Mode 3, tty/text mode data transfers
                                <1>       ;          BL =
                                <1>       ;            Bits 0&1, Transfer direction
                                <1>       ;                  0 - System to system
                                <1>       ;                  1 - User to system
                                <1>       ;                  2 - System to user
                                <1>       ;                  3 - Exhange (Swap) - 28/01/2021
                                <1>       ;            Bits 2, Transfer Type
                                <1>       ;                  0 - Display page (complete) transfer
                                <1>       ;                  1 - Display page window (col,row) transfer
                                <1>       ;            ; 28/01/2021
```

```
5782                    <1>     ;           Bits 3..7 - Reserved, undefined (must be 0)
5783                    <1>     ;            ; 28/01/2021
5784                    <1>     ;         /// BL = 0 -> System to system (display page) transfer
5785                    <1>     ;            CL = Source page (0FFh = current video page)
5786                    <1>     ;            DL = Destination page (0FFh = current video page)
5787                    <1>     ;            (Note: Nothing to do if src & dest are same page)
5788                    <1>     ;         /// BL = 1&2 -> user to system & system to user transfer
5789                    <1>     ;            ECX = User's buffer address
5790                    <1>     ;            DL = Video page (0FFh = current video page)
5791                    <1>     ;         /// BL = 3 -> exchange (swap) display page ; 28/01/2021
5792                    <1>     ;            ECX = User's buffer address
5793                    <1>     ;            DL = Video page (0FFh = current video page)
5794                    <1>     ;            EDI = Swap address in user's memory (must be > 0)
5795                    <1>     ;         /// BL = 5&6&7 -> user to system, system to user transfer
5796                    <1>     ;           (system window is in current/active display page)
5797                    <1>     ;            ESI = User's buffer address
5798                    <1>     ;            ECX Low 16 bits = Top left column (X1 position)
5799                    <1>     ;            ECX High 16 bits = Top row (Y1 position)
5800                    <1>     ;            EDX Low 16 bits = Bottom right column (X2 position)
5801                    <1>     ;            EDX High 16 bits = Bottom row (Y2 position)
5802                    <1>     ;               If BL = 5 or BL bit 0 & bit 1 are 1 ; 28/01/2021
5803                    <1>     ;            EDI = Swap address (in user's memory space)
5804                    <1>     ;            (If swap address > 0, previous content of the window
5805                    <1>     ;            will be saved into swap area in user's memory space)
5806                    <1>     ;         /// BL = 4 -> system to system transfer
5807                    <1>     ;            ESI = System's source buffer (video page) address
5808                    <1>     ;            ECX Low 16 bits = Top left column (X1 position)
5809                    <1>     ;            ECX High 16 bits = Top row (Y1 position)
5810                    <1>     ;            EDX Low 16 bits = Bottom right column (X2 position)
5811                    <1>     ;            EDX High 16 bits = Bottom row (Y2 position)
5812                    <1>     ;            EDI = System's destination buffer (video page) addr
5813                    <1>     ;
5814                    <1>     ;            ; 06/02/2021
5815                    <1>     ;            ; 05/02/2021
5816                    <1>     ;            ; 01/02/2021, 02/02/2021
5817                    <1>     ;            ; 30/01/2021, 31/02/2021
5818                    <1>     ;            ; 29/01/2021 (major modification)
5819                    <1>     ;            ; 23/11/2020 (major modification)
5820                    <1>     ;            ; 22/11/2020 (bugfixes and extensions)
5821                    <1>     ;       BH = 1 = VGA Graphics (0A0000h) data transfers
5822                    <1>     ;              BL bit 7
5823                    <1>     ;                 resolution (screen width) option
5824                    <1>     ;                 0 = 320 pixels
5825                    <1>     ;                 1 = 640 pixels
5826                    <1>     ;              .. followings are same with SVGA transfer function
5827                    <1>     ;              BL bit 6
5828                    <1>     ;                 direction option
5829                    <1>     ;                 0 = user to system (video memory)
5830                    <1>     ;                 1 = system to user
5831                    <1>     ;              BL bit 5
5832                    <1>     ;                 masked/direct (non-masked) operations
5833                    <1>     ;                 1 = masked, 0 = non-masked (direct)
5834                    <1>     ;              BL bit 4
5835                    <1>     ;                 page/window option
5836                    <1>     ;                 1 = window, 0 = display page (screen)
5837                    <1>     ;              BL bit 0 to 3 (pixel operation types)
5838                    <1>     ;                 0 = Copy pixels (colors) ((mask color))
5839                    <1>     ;                 1 = Change (New, Fill) color
5840                    <1>     ;                 2 = Add color
5841                    <1>     ;                 3 = Sub color
5842                    <1>     ;                 4 = OR color
5843                    <1>     ;                 5 = AND color
5844                    <1>     ;                 6 = XOR color
5845                    <1>     ;                 7 = NOT color
5846                    <1>     ;                 8 = NEG color
5847                    <1>     ;                 9 = INC color
5848                    <1>     ;                10 = DEC color
5849                    <1>     ;                11 = Mix (Average) colors
5850                    <1>     ;                12 = Replace pixel colors
5851                    <1>     ;                13 = Copy pixel block(s)
5852                    <1>     ;                14 = Write line(s)
5853                    <1>     ;                15 = Write character (font)
5854                    <1>     ;
5855                    <1>     ;           Input Registers for pixel operations:
5856                    <1>     ;                Same with LFB data transfer function below
5857                    <1>     ;
5858                    <1>     ;               ; 25/02/2021
5859                    <1>     ;               ; 05/02/2021, 06/02/2021
5860                    <1>     ;               ; 01/02/2021, 02/02/2021
5861                    <1>     ;               ; 30/01/2021, 31/02/2021
5862                    <1>     ;               ; 29/01/2021 (major modification)
5863                    <1>     ;               ; 23/11/2020 (major modification)
5864                    <1>     ;               ; 22/11/2020 (bugfixes and extensions)
5865                    <1>     ;         BH = 2 = Super VGA, LINEAR FRAME BUFFER data transfers
5866                    <1>     ;               BL bit 7
5867                    <1>     ;                  unused (invalid), must be 0
5868                    <1>     ;               BL bit 6
5869                    <1>     ;                  direction option
5870                    <1>     ;                  0 = user to system (video memory)
5871                    <1>     ;                  1 = system to user
5872                    <1>     ;               BL bit 5
5873                    <1>     ;                  masked/direct (non-masked) operations
5874                    <1>     ;                  1 = masked, 0 = non-masked (direct)
5875                    <1>     ;               BL bit 4
5876                    <1>     ;                  page/window option
5877                    <1>     ;                  1 = window, 0 = display page (screen)
5878                    <1>     ;               BL bit 0 to 3 (pixel operation types)
5879                    <1>     ;                  0 = Copy pixels (colors) ((mask color))
5880                    <1>     ;                  1 = Change (New, Fill) color
5881                    <1>     ;                  2 = Add color
5882                    <1>     ;                  3 = Sub color
5883                    <1>     ;                  4 = OR color
5884                    <1>     ;                  5 = AND color
5885                    <1>     ;                  6 = XOR color
5886                    <1>     ;                  7 = NOT color
```

```
5887                    <1>     ;               8 = NEG color
5888                    <1>     ;               9 = INC color
5889                    <1>     ;              10 = DEC color
5890                    <1>     ;              11 = Mix (Average) colors
5891                    <1>     ;              12 = Replace pixel colors
5892                    <1>     ;              13 = Copy pixel block(s)
5893                    <1>     ;              14 = Write line(s)
5894                    <1>     ;              15 = Write character (font)
5895                    <1>     ;
5896                    <1>     ;           Note: If HW of EBX > 0, it is VESA VBE mode number
5897                    <1>     ;                 otherwise, function will be applied
5898                    <1>     ;                 to current (VESA VBE) video mode.
5899                    <1>     ;
5900                    <1>     ;           Input Registers for pixel operations:
5901                    <1>     ;              -- user to system & system to system --
5902                    <1>     ;              -- (BL = 0 to 0Fh) -- non-masked, screen --
5903                    <1>     ;              -- (BL = 10h to 1Fh) -- non-masked, window --
5904                    <1>     ;              -- (BL = 20h to 2Fh) -- masked, screen --
5905                    <1>     ;              -- (BL = 30h to 3Fh) -- masked, window --
5906                    <1>     ;              (*) window, (**) masked (***) sys to sys
5907                    <1>     ;              for BL bit 0 to 3
5908                    <1>     ;              00h: COPY PIXELS
5909                    <1>     ;                If BL bit 4 = 0 ; 21/02/2021
5910                    <1>     ;                   full screen copy
5911                    <1>     ;                   ECX & EDX will not be used
5912                    <1>     ;                  (user buffer must fit to display page)
5913                    <1>     ;                If BL bit 4 = 1 ; 21/02/2021
5914                    <1>     ;                  ECX = start position (row, column) (*)
5915                    <1>     ;                       (HW = row, CX = column)
5916                    <1>     ;                  EDX = size (rows, colums) (*)
5917                    <1>     ;                       (HW = rows, DX = columns)
5918                    <1>     ;                         (0 -> invalid)
5919                    <1>     ;                         (1 -> horizontal or vertical line)
5920                    <1>     ;                  ESI = user's buffer address
5921                    <1>     ;                  EDI = mask color (**) ; 25/02/2021
5922                    <1>     ;                       (this color will be excluded)
5923                    <1>     ;              01h: CHANGE PIXEL COLORS
5924                    <1>     ;              02h: ADD PIXEL COLORS
5925                    <1>     ;              03h: SUB PIXEL COLORS
5926                    <1>     ;              04h: OR PIXEL COLORS
5927                    <1>     ;              05h: AND PIXEL COLORS
5928                    <1>     ;              06h: XOR PIXEL COLORS
5929                    <1>     ;              0Bh: MIX PIXEL COLORS
5930                    <1>     ;                   CL = color (8 bit, 256 colors)
5931                    <1>     ;                  ECX = color (16 bit and true colors)
5932                    <1>     ;                  EDX = start position (row, column) (*)
5933                    <1>     ;                      (HW = row, DX = column)
5934                    <1>     ;                  ESI = size (rows, colums) (*)
5935                    <1>     ;                       (HW = rows, SI = columns)
5936                    <1>     ;                  EDI = mask color (**) ; 25/02/2021
5937                    <1>     ;                       (this color will be excluded)
5938                    <1>     ;              07h: NOT PIXEL COLORS
5939                    <1>     ;              08h: NEG PIXEL COLORS
5940                    <1>     ;              09h: INC PIXEL COLORS
5941                    <1>     ;              0Ah: DEC PIXEL COLORS
5942                    <1>     ;                  ECX = start position (row, column) (*)
5943                    <1>     ;                       (HW = row, CX = column)
5944                    <1>     ;                  EDX = size (rows, colums) (*)
5945                    <1>     ;                       (HW = rows, DX = columns)
5946                    <1>     ;                         (0 -> invalid)
5947                    <1>     ;                         (1 -> horizontal or vertical line)
5948                    <1>     ;                  EDI = mask color (**) ; 25/02/2021
5949                    <1>     ;                       (this color will be excluded)
5950                    <1>     ;              0Ch: REPLACE PIXEL COLORS
5951                    <1>     ;                   CL = current color (8 bit, 256 colors)
5952                    <1>     ;                  ECX = current color (16 bit and true colors)
5953                    <1>     ;                   DL = new color (8 bit, 256 colors)
5954                    <1>     ;                  EDX = new color (16 bit and true colors)
5955                    <1>     ;                  ESI = start position (row, column) (*)
5956                    <1>     ;                       (HW = row, SI = column)
5957                    <1>     ;                  EDI = size (rows, colums) (*)
5958                    <1>     ;                       (HW = rows, DI = columns)
5959                    <1>     ;              0Dh: COPY PIXEL BLOCK(S) -full screen-
5960                    <1>     ;                 -If BL bit 5 is 0-
5961                    <1>     ;                  ECX = start position (row, column) (*)
5962                    <1>     ;                       (HW = row, CX = column)
5963                    <1>     ;                  EDX = size (rows, colums) (*)
5964                    <1>     ;                       (HW = rows, DX = columns)
5965                    <1>     ;                         (0 -> invalid)
5966                    <1>     ;                         (1 -> horizontal or vertical line)
5967                    <1>     ;                  ESI = destination (row, column) (***)
5968                    <1>     ;                 -If BL bit 5 is 1-
5969                    <1>     ;                       CL = color (8 bit, 256 colors)
5970                    <1>     ;                    ECX = color (16 bit and true colors)
5971                    <1>     ;                    EDX = count of blocks (not bytes)
5972                    <1>     ;                         (limit: 2048 blocks)
5973                    <1>     ;                    ESI = user's buffer address
5974                    <1>     ;                       contains 64 bits block data
5975                    <1>     ;                       BLOCK ADDRESS - (row, col), dword
5976                    <1>     ;                       (first 32 bits)
5977                    <1>     ;                       BLOCK SIZE - (rows, cols), dword
5978                    <1>     ;                       (second 32 bits)
5979                    <1>     ;              ; 10/02/2021
5980                    <1>     ;              0Eh: WRITE LINE(s) -full screen-
5981                    <1>     ;                 -If BL bit 5 is 0-
5982                    <1>     ;                  CL = color (8 bit, 256 colors)
5983                    <1>     ;                 ECX = color (16 bit and true colors)
5984                    <1>     ;                  DX = low 12 bits - size (length)
5985                    <1>     ;                      high 4 bits - direction or type
5986                    <1>     ;                          0 - Horizontal line
5987                    <1>     ;                          1 - Vertical line
5988                    <1>     ;                        > 1 - undefined, invalid
5989                    <1>     ;                 ESI = start position (row, column)
5990                    <1>     ;                      (HW = row, SI = column)
5991                    <1>     ;                 -If BL bit 5 is 1-
```

```
5992                    <1>     ;               CL = color (8 bit, 256 colors)
5993                    <1>     ;              ECX = color (16 bit and true colors)
5994                    <1>     ;               DX = number of lines (in user buffer)
5995                    <1>     ;                    (limit: 2048 lines)
5996                    <1>     ;              ESI = user's buffer
5997                    <1>     ;                    contains 64 bit data for lines
5998                    <1>     ;                    START POINT: 32 bit (row, col)
5999                    <1>     ;                    LENGTH: 32 bit
6000                    <1>     ;                    high 16 bits - 0
6001                    <1>     ;                    bit 0-11 - length
6002                    <1>     ;                    bit 12-15 - type (length)
6003                    <1>     ;           0Fh: WRITE CHARACTER (FONT)
6004                    <1>     ;                CL = char's color (8 bit, 256 colors)
6005                    <1>     ;               ECX = char's color (16 bit and true colors)
6006                    <1>     ;                DL = Character's ASCII code
6007                    <1>     ;                DH bit 0 -> font height
6008                    <1>     ;                   0 -> 8x16 character font
6009                    <1>     ;                   1 -> 8x8 character font
6010                    <1>     ;                DH bit 1 & 2 -> scale
6011                    <1>     ;                   0 = 1/1 (8 pixels per char row)
6012                    <1>     ;                   1 = 2/1 (16 pixels per char row)
6013                    <1>     ;                   2 = 3/1 (24 pixels per char row)
6014                    <1>     ;                   3 = 4/1 (32 pixels per char row)
6015                    <1>     ;                DH bit 6 -> [ufont] option (1 = use [ufont])
6016                    <1>     ;                If DH bit 7 = 1
6017                    <1>     ;                  USER FONT (from user buffer)
6018                    <1>     ;                     DL = 0 -> 8x8 (width: 1 byte per row)
6019                    <1>     ;                     DL = 1 -> 8x16
6020                    <1>     ;                     DL = 2 -> 16x16 (width: 2 bytes)
6021                    <1>     ;                     DL = 3 -> 16x32
6022                    <1>     ;                     DL = 4 -> 24x24 (width: 3 bytes)
6023                    <1>     ;                     DL = 5 -> 24x48
6024                    <1>     ;                     DL = 6 -> 32x32 (width: 4 bytes)
6025                    <1>     ;                     DL = 7 -> 32x64
6026                    <1>     ;                     DL > 7 -> invalid (unused)
6027                    <1>     ;                  EDI = user's font buffer address
6028                    <1>     ;                     (NOTE: byte order is as row0,row1,row2..)
6029                    <1>     ;               ESI = start position (row, column) (*)
6030                    <1>     ;                      (HW = row, SI = column)
6031                    <1>     ;
6032                    <1>     ;           -- system to user --
6033                    <1>     ;              BL (bit 0 to 7)
6034                    <1>     ;           40h: COPY PIXELS (full screen, display page)
6035                    <1>     ;               EDI = user's buffer address
6036                    <1>     ;           41h: COPY PIXELS (window)
6037                    <1>     ;               ECX = start position (row, column) (*)
6038                    <1>     ;                    (HW = row, CX = column)
6039                    <1>     ;               EDX = size (rows, colums) (*)
6040                    <1>     ;                    (HW = rows, DX = columns)
6041                    <1>     ;                     (<=1 -> horizontal or vertical line)
6042                    <1>     ;               EDI = user's buffer address
6043                    <1>     ;
6044                    <1>     ;           Example: (29/01/2021)
6045                    <1>     ;               ecx = 00400064h (start at row 64, column 100)
6046                    <1>     ;               edx = 00320048h (size: 50 rows, 72 columns)
6047                    <1>     ;                             (end at row 114, column 172)
6048                    <1>     ;               If video memory starts at 0A0000h
6049                    <1>     ;               and if resolution is 320x200 (256 colors) ..
6050                    <1>     ;               window start offset: (64*320)+100 = 20580
6051                    <1>     ;               window size: 16072 bytes (pixels)
6052                    <1>     ;               window end offset: 20580+16072 = 36652
6053                    <1>     ;               window start address: 0A0000h+564h = 0A5064h
6054                    <1>     ; Outputs:
6055                    <1>     ;     EAX = transfer/byte count
6056                    <1>     ;
6057                    <1>     ;     NOTE: If the source or destination address passes out of
6058                    <1>     ;     video pages (display memory limits), data will not be transferred
6059                    <1>     ;     and EAX will return as 0.
6060                    <1>     ;
6061                    <1>     ; 08/02/2021
6062                    <1>     ; 07/02/2021
6063                    <1>     ; 04/01/2021
6064                    <1>     ; PIXEL READ/WRITE (in current/active video mode)
6065                    <1>     ;
6066                    <1>     ;     BH = 3 = Read/Write pixel(s) -for all graphics modes-
6067                    <1>     ;          BL =
6068                    <1>     ;            0 = Read pixel
6069                    <1>     ;            1 = Write pixel
6070                    <1>     ;            2 = swap pixel colors
6071                    <1>     ;            3 = mix pixel colors
6072                    <1>     ;          29/01/2021
6073                    <1>     ;            4 = read pixels from user defined positions
6074                    <1>     ;            5 = write single color pixels to user defined positions
6075                    <1>     ;            6 = write multi color pixels to user defined positions
6076                    <1>     ;
6077                    <1>     ;           > 6 = invalid/unimplemented
6078                    <1>     ;
6079                    <1>     ;          .. for BL = 0 to 5
6080                    <1>     ;          CL = color for writing pixel(s) or
6081                    <1>     ;          ECX = color for writing pixel(s) in true color modes
6082                    <1>     ;
6083                    <1>     ;          EDX = Offset from start of video memory (0A0000h)
6084                    <1>     ;               or start of linear frame buffer
6085                    <1>     ;
6086                    <1>     ;          07/02/2021
6087                    <1>     ;          .. for BL = 4
6088                    <1>     ;          EDI = user's destination buffer address for pixel colors
6089                    <1>     ;          29/01/2021
6090                    <1>     ;          .. for BL = 4 & 5
6091                    <1>     ;          ESI = user's source buffer address for BL = 4 & 5
6092                    <1>     ;          (buffer contains dword offset positions for pixels)
6093                    <1>     ;          EDX = number of pixels
6094                    <1>     ;          .. for BL = 6
6095                    <1>     ;          ESI = user's buffer address for BL = 6
6096                    <1>     ;          (buffer contains dword offset position and dword color
```

```
6097                    <1>     ;           value for each pixel)
6098                    <1>     ;          EDX = number of pixels
6099                    <1>     ;
6100                    <1>     ; Note:
6101                    <1>     ;     Pixel read/write will be performed in current video mode.
6102                    <1>     ;     If [CRT_MODE] < 0FFh, 0A0000h will be used
6103                    <1>     ;        as video memory and limit will be 65536
6104                    <1>     ;        (new/mix pixel color will be in CL)
6105                    <1>     ;     if [CRT_MODE] = 0FFh (VESA VBE video mode)
6106                    <1>     ;        LFB base address will be used as video memory
6107                    <1>     ;        and limit will be video page size
6108                    <1>     ;        (new/mix pixel color will be in CL)
6109                    <1>     ;
6110                    <1>     ; Outputs:
6111                    <1>     ;     EAX = pixel color (according to BL and ECX input)
6112                    <1>     ;     EAX = 0 (pixel color is 0 or there is an error)
6113                    <1>     ;          (BL will return as 0FFh if there is an error)
6114                    <1>     ;     ; 29/01/2021
6115                    <1>     ;     EAX = number of pixels (for BL input = 4&5&6)
6116                    <1>     ;
6117                    <1>     ; DIRECT (STANDARD VGA/CGA) DISPLAY MEMORY ACCESS FUNCTIONS:
6118                    <1>     ;
6119                    <1>     ;     BH = 4 = CGA direct video memory (0B8000h, 32K) access
6120                    <1>     ;            Page directory & page tables of the user's
6121                    <1>     ;            program will be updated to direct access to
6122                    <1>     ;            0B8000h (32K) video (CGA, color) memory; if
6123                    <1>     ;            there is not a permission conflict or lock!
6124                    <1>     ;              (User's program/process will have permission to
6125                    <1>     ;            access locked display memory if the owner is
6126                    <1>     ;            it's parent.)
6127                    <1>     ;
6128                    <1>     ;         Screen width = 320
6129                    <1>     ;
6130                    <1>     ;     BH = 5 = VGA direct video memory (0A0000h, 64K) access
6131                    <1>     ;            Page directory & page tables of the user's
6132                    <1>     ;            program will be updated to direct access to
6133                    <1>     ;            0A0000h (64K) video (VGA) memory; if there is not
6134                    <1>     ;            a permission conflict or lock!
6135                    <1>     ;              (User's program/process will have permission to
6136                    <1>     ;            access locked display memory if the owner is
6137                    <1>     ;            it's parent.)
6138                    <1>     ;
6139                    <1>     ;         ; 23/11/2020
6140                    <1>     ;         Screen width options = 320, 640, 800
6141                    <1>     ;
6142                    <1>     ; Outputs:
6143                    <1>     ;     EAX = Display memory address for direct access
6144                    <1>     ;           0A0000h for VGA, 0B8000h for CGA
6145                    <1>     ;     (Display memory size: 32K for CGA, 64K for VGA)
6146                    <1>     ;     EAX = 0 if display page access permission has been denied.
6147                    <1>     ;           (Locked!)
6148                    <1>     ;
6149                    <1>     ; LINEAR FRAME BUFFER ACCESS FUNCTIONS:
6150                    <1>     ;
6151                    <1>     ;     BH = 6 = Linear Frame Buffer direct video memory access
6152                    <1>     ;
6153                    <1>     ;            Page directory & page tables of the user's
6154                    <1>     ;            program will be updated to direct access to
6155                    <1>     ;            the configured LFB (Linear Frame Buffer) address,
6156                    <1>     ;            if there is not a permission conflict or lock!
6157                    <1>     ;              (User's program/process will have permission to
6158                    <1>     ;            access locked display memory if the owner is
6159                    <1>     ;            it's parent.)
6160                    <1>     ;
6161                    <1>     ;        ; 10/12/2020
6162                    <1>     ;        BL = 0FFh -> Direct LFB access for current video mode
6163                    <1>     ;        BL = XXh < 0FFh -> Direct LFB access
6164                    <1>     ;                          for VESA video mode 1XXh
6165                    <1>     ;
6166                    <1>     ;            Return: EAX = Linear Frame Buffer address
6167                    <1>     ;                   (EAX = 0 -> error)
6168                    <1>     ;                 If EAX > 0
6169                    <1>     ;                   EDX = Frame Buffer Size in bytes
6170                    <1>     ;                     BH = Requested Video Mode - 100h
6171                    <1>     ;                          (VESA VBE video modes)
6172                    <1>     ;                     BL = bits per pixel
6173                    <1>     ;                         8 = 256 colors, 8
6174                    <1>     ;                          16 = 65536 colors, 5-6(G)-5
6175                    <1>     ;                          24 = RGB, 16M colors, 8-8-8
6176                    <1>     ;                          32 = RGB + alpha bytes, 8-8-8-8
6177                    <1>     ;                 If BH = 0FFh
6178                    <1>     ;                    BL = VGA/CGA video mode (also EAX = 0)
6179                    <1>     ;
6180                    <1>     ;            ** Function will return with EAX = 0 if the mode
6181                    <1>     ;            is not a valid VESA VBE video mode as 1??h **
6182                    <1>     ;
6183                    <1>     ;            ECX = Pixel resolution
6184                    <1>     ;                  CX = Width (320, 640, 800, 1024, 1366, 1920)
6185                    <1>     ;                  High 16 bits of ECX = Height
6186                    <1>     ;
6187                    <1>     ;     23/11/2020
6188                    <1>     ; ***  GET VIDEO MODE & LINEAR FRAME BUFFER INFO
6189                    <1>     ;     (This function -7- also is used for VGA and CGA modes)
6190                    <1>     ;
6191                    <1>     ;     BH = 7 = Get Linear Frame Buffer info
6192                    <1>     ;
6193                    <1>     ;        ; 22/01/2021
6194                    <1>     ;        ; 10/12/2020
6195                    <1>     ;        BL = any -not used- (22/01/2021)
6196                    <1>     ;
6197                    <1>     ;            Return:
6198                    <1>     ;            EAX = Frame Buffer Address (0 = is not in use)
6199                    <1>     ;            EDX = Frame Buffer Size in bytes
6200                    <1>     ;             BH = Current Video Mode - 100h  ; 22/01/2021
6201                    <1>     ;                 (VESA VBE video modes)
```

```
6202                    <1>     ;               BL = bits per pixel
6203                    <1>     ;                    8 = 256 colors, 8
6204                    <1>     ;                    16 = 65536 colors, 5-6(G)-5
6205                    <1>     ;                    24 = RGB, 16M colors, 8-8-8
6206                    <1>     ;                    32 = RGB + alpha bytes, 8-8-8-8
6207                    <1>     ;               If BH = 0FFh
6208                    <1>     ;                  BL = VGA/CGA video mode (also EAX = 0)
6209                    <1>     ;
6210                    <1>     ;               Note:
6211                    <1>     ;               Alpha byte will be used as virtual color index.
6212                    <1>     ;               (32 bit pixel colors.. byte 0,1,2 rgb and 3 alpha)
6213                    <1>     ;
6214                    <1>     ;               ** Function will return with EAX = 0 if the mode
6215                    <1>     ;               is not a valid VESA VBE video mode as 1??h **
6216                    <1>     ;
6217                    <1>     ;               ECX = Pixel resolution
6218                    <1>     ;                     CX = Width (320, 640, 800, 1024, 1366, 1920)
6219                    <1>     ;                     High 16 bits of ECX = Height
6220                    <1>     ;
6221                    <1>     ;       NOTE: Each process will have it's own frame buffer
6222                    <1>     ;             address and resolution parameters in 'u' area.
6223                    <1>     ;             Then, if the current frame buffer & resolution
6224                    <1>     ;             is different, frame buffer r/w functions
6225                    <1>     ;             will use scale factor to convert process's
6226                    <1>     ;               pixel coordinates to actual screen coordinates.
6227                    <1>     ;             resolution -> dimensional scale
6228                    <1>     ;             color size -> color scale
6229                    <1>     ;            * RGB (TRUE) colors to 256 colors conversion:
6230                    <1>     ;              TRUE Colors -> 8,8,8 (R,G,B; byte 0 is R)
6231                    <1>     ;             256 colors -> 2,2,2,2 (R,G,B,L; bit 0&1 is R)
6232                    <1>     ;                bit 6&7 -> luminosity base level (0,1,2,3)
6233                    <1>     ;                bit 4&5 -> blue level (0,1,2,3)
6234                    <1>     ;                bit 2%3 -> green level (0,1,2,3)
6235                    <1>     ;                   bit 0&1 -> red level (0,1,2,3)
6236                    <1>     ;               Example: total red level : luminosity + red level
6237                    <1>     ;               Luminosity base level: 0 -> 16
6238                    <1>     ;                                      1 -> 32
6239                    <1>     ;                                      2 -> 64
6240                    <1>     ;                                      3 -> 128
6241                    <1>     ;               Color level:
6242                    <1>     ;                                     0 -> 0
6243                    <1>     ;                                     1 -> luminosity level
6244                    <1>     ;                                     2 -> luminosity level + 64
6245                    <1>     ;                                     3 -> 255
6246                    <1>     ;               Luminosity base level = min (R,G,B)
6247                    <1>     ;                       if it is <16, it will be set to 16
6248                    <1>     ;               Color levels: Color values are fixed to (nearest)
6249                    <1>     ;                   one of all possible set level (step) values
6250                    <1>     ;                   (according to luminosity base level); then
6251                    <1>     ;                   color levels are set to R-L, G-L, B-L.
6252                    <1>     ;             For example: If luminosity base level is 32
6253                    <1>     ;                   all possible set values are 0, 32, 96, 255.
6254                    <1>     ;
6255                    <1>     ;               * RGB (TRUE) colors to 16 colors conversion:
6256                    <1>     ;               16 colors: R,B,G,L bits (4 bits)
6257                    <1>     ;                     If any one of R,G,B >= 128 L = 1
6258                    <1>     ;                     If max. value of (R,G,B) >= 32, it is 1
6259                    <1>     ;                        else all color bits (R&G&B&L) are 0
6260                    <1>     ;                     If the second value >= max. value / 2
6261                    <1>     ;                        it is 1
6262                    <1>     ;                     If third value value >= max. value / 2
6263                    <1>     ;                        it is 1
6264                    <1>     ;               Example: R = 132, G = 64, B = 78
6265                    <1>     ;                      L = 1, R = 1
6266                    <1>     ;                      G < 66 --> G = 0
6267                    <1>     ;                      B >= 66 --> B = 1
6268                    <1>     ;
6269                    <1>     ; 10/12/2020
6270                    <1>     ; SET VIDEO MODE (& RETURN LFB INFO for VESA VBE VIDEO MODES)
6271                    <1>     ;
6272                    <1>     ;       BH = 8 = Set Video Mode
6273                    <1>     ;
6274                    <1>     ;               BL = Requested Video Mode (method)
6275                    <1>     ;               If BL = 0FFh
6276                    <1>     ;                  CX = VESA VBE Video Mode
6277                    <1>     ;                  ; 11/12/2020
6278                    <1>     ;                  If EDX > 0 -> LFB INFO (user) buffer addr
6279                    <1>     ;               If BL < 0FFh, it is VGA/CGA video mode and
6280                    <1>     ;                     CX & EDX will not be used
6281                    <1>     ;
6282                    <1>     ;            NOTE: The last VESA VBE video mode is 11Bh but
6283                    <1>     ;            TRDOS 386 will permit to set video mode upto 11Fh.
6284                    <1>     ;            Above 11Fh, from 140h to 1FEh, it will be accepted
6285                    <1>     ;            as Bochs/Plex86 emulator video mode and it will be
6286                    <1>     ;            used only if [vbe3] = 2 and detected
6287                    <1>     ;            video bios is BOCHS/PLEX86/QEMU/VIRTUALBOX vbios.
6288                    <1>     ;
6289                    <1>     ; Outputs:
6290                    <1>     ;       EAX = Requested (Proper) video mode number + 1
6291                    <1>     ;             ("dec eax" by user will give requested video mode),
6292                    <1>     ;
6293                    <1>     ;       If BL input is 0FFh
6294                    <1>     ;          EDX = LFBINFO table/structure (in user's buffer addr)
6295                    <1>     ;          EDX = 0 -> Invalid LFBINFO (do not use it)
6296                    <1>     ;
6297                    <1>     ;       EAX = 0 -> Error (but EDX will not be changed)
6298                    <1>     ;
6299                    <1>     ; 03/12/2020
6300                    <1>     ; VESA VBE3 VIDEO BIOS (32 bit) PROTECTED MODE INTERFACE SETTINGS
6301                    <1>     ;
6302                    <1>     ;       BH = 9 = set/get VBE3 Protected Mode Interface parameters
6303                    <1>     ;
6304                    <1>     ;               BL = 0 - Disable protected mode interface
6305                    <1>     ;                      ([pmi32] = 0)
6306                    <1>     ;                      Return: AL = 1
```

```
6307  <1>       ;           BL = 1 - Enable protected mode Interface
6308  <1>       ;                  ([pmi32] = 1)
6309  <1>       ;                  Return: AL = 2
6310  <1>       ;           BL = 2 - Get protected mode interface status
6311  <1>       ;                  Return: AL = [pmi32] + 1 (AL = 1 or 2)
6312  <1>       ;
6313  <1>       ;           If [vbe3] <> 3 --> AL = 0
6314  <1>       ;
6315  <1>       ;           ; 17/01/2021
6316  <1>       ;           BL = 3 - Disable/Cancel restore permission to user
6317  <1>       ;                  Return: AL = 1 (if disabled) or 0
6318  <1>       ;           BL = 4 - Enable/Give restore permission to user
6319  <1>       ;                  Return: AL = 2 (if enabled) or 0
6320  <1>       ;           BL = 5 - Get video state save/restore status
6321  <1>       ;                   (permission status)
6322  <1>       ;                   Return: AL = Status (enabled = 1)
6323  <1>       ;                        ; 22/01/2021
6324  <1>       ;                        AH = state options ([srvso])
6325  <1>       ;           BL = 6 - Return VESA VBE number/status
6326  <1>       ;                   Return: AX = status
6327  <1>       ;                       if AH =  2, AL > 0 : Emulator
6328  <1>       ;                         AH =  3, VESA VBE3 video bios
6329  <1>       ;           ; 28/02/2021
6330  <1>       ;           BL = 7 - Set true color mode as 32bpp (default)
6331  <1>       ;                  Return: AX = 32 (if VBE3)
6332  <1>       ;           NOTE: Initial/default value is 32bpp for vbe3.
6333  <1>       ;                  Return: AX = 0 -> error
6334  <1>       ;           BL = 8 - Set true color mode as 24bpp (default)
6335  <1>       ;                  Return: AX = 24
6336  <1>       ;                  ;Return: AX = 0 -> error
6337  <1>       ;           BL = 9 - Return default/current true color mode
6338  <1>       ;                  Return: AX = 32 (32 bpp)
6339  <1>       ;                  Return: AX = 24 (24 bpp)
6340  <1>       ;                  Return: AX = 0 -> error (not VESA bios)
6341  <1>       ;
6342  <1>       ;           BL > 9 : not implemented (28/02/2021)
6343  <1>       ;
6344  <1>       ;     ; 19/01/2021 ([u.uid] check)
6345  <1>       ;     Note: Enabling/Disabling are done by root ([u.uid] = 0)
6346  <1>       ;           while [multi_tasking] = 0.
6347  <1>       ;
6348  <1>       ; 23/12/2020
6349  <1>       ; VIDEO MEMORY MAPPING:
6350  <1>       ;     BH = 10 = Map video memory to user's buffer
6351  <1>       ;
6352  <1>       ;        BL = 0 : CGA memory (0B8000h) map (32K)
6353  <1>       ;        BL = 1 : VGA memory (0A0000h) map (64K)
6354  <1>       ;        BL = 2 : SVGA memory (LFB) map to user's buffer
6355  <1>       ;
6356  <1>       ;     ECX = User's buffer addr (low 12 bits will be cleared)
6357  <1>       ;     EDX = Buffer size in bytes (if BL = 2)
6358  <1>       ;           (will be trimmed if LFB size < EDX)
6359  <1>       ; Return:
6360  <1>       ;     EAX = physical address of video memory (buffer)
6361  <1>       ;     EBX = mapped (actual) size of video memory (bytes)
6362  <1>       ;     ECX = virtual start address of user's video buffer
6363  <1>       ;     EDX is same with EDX input
6364  <1>       ;
6365  <1>       ;     (Note: Memory page boundaries will be applied
6366  <1>       ;      to buffer size and buff start addr by rounding down.
6367  <1>       ;      Rounded size & address values must not be zero.)
6368  <1>       ;     -Normally, it is expected to request mapping by using
6369  <1>       ;      correct buffer size of current or desired video mode-
6370  <1>       ;
6371  <1>       ;     EAX = 0 -> error ! memory can not mapped to user
6372  <1>       ;
6373  <1>       ; 04/01/2021
6374  <1>       ; SET/READ COLOR PALETTE (set/read DAC color registers)
6375  <1>       ;     ((256 colors (8bpp) VGA/CGA video hardware feature))
6376  <1>       ;
6377  <1>       ;     BH = 11 = Set/Read DAC color registers
6378  <1>       ;
6379  <1>       ;        (BL<4 Original method for std VGA video hardware)
6380  <1>       ;        BL = 0 : Read all DAC color registers (256 colors)
6381  <1>       ;            (6 bit colors, in RGB order)
6382  <1>       ;        BL = 1 : Set all DAC color registers (256 colors)
6383  <1>       ;            (6 bit colors, in RGB order)
6384  <1>       ;        BL = 2 : Read single DAC color register
6385  <1>       ;            (6 bit color, in RGB order)
6386  <1>       ;           ((EAX will return with color value))
6387  <1>       ;           CL = DAC color register (index)
6388  <1>       ;        BL = 3 : Set/Write single DAC color register
6389  <1>       ;            (6 bit color, in RGB order, bit 6&7 are 0)
6390  <1>       ;           ECX byte 0 - DAC color register
6391  <1>       ;           ECX byte 1 - Red (6 bit)
6392  <1>       ;           ECX byte 2 - Green (6 bit)
6393  <1>       ;           ECX byte 3 - Blue (6 bit)
6394  <1>       ;        (BL>3 Alternative method for BMP files etc.)
6395  <1>       ;        BL = 4 : Read all DAC color registers (256 colors)
6396  <1>       ;            (8 bit colors, in BGR order, bit 0&1 is 0)
6397  <1>       ;        BL = 5 : Set all DAC color registers (256 colors)
6398  <1>       ;            (8 bit colors, in BGR order, bit 0&1 is 0)
6399  <1>       ;        BL = 6 : Read single DAC color register
6400  <1>       ;            (8 bit color, in BGR order, bit 0&1 is 0)
6401  <1>       ;           ((EAX will return with color value))
6402  <1>       ;           CL = DAC color register (index)
6403  <1>       ;        BL = 7 : Set/Write single DAC color register
6404  <1>       ;            (8 bit color, bit 0&1 are 0)
6405  <1>       ;           ECX byte 0 - DAC color register
6406  <1>       ;           ECX byte 1 - Blue (8 bit)
6407  <1>       ;           ECX byte 2 - Green (8 bit)
6408  <1>       ;           ECX byte 3 - Red (8 bit)
6409  <1>       ;
6410  <1>       ;       BL > 7 : invalid (not implemented)
6411  <1>       ;
```

```
6412 <1>       ;   if BL bit 2 is 1, 6 bit colors converted to 8 bit colors
6413 <1>       ;      (low two bits of color bytes will be 0)
6414 <1>       ;     ((color byte 00111111b will be converted to 11111100b))
6415 <1>       ;     and RGB byte order will be
6416 <1>       ;           byte 0 - Blue (low 2 bits are 0)
6417 <1>       ;           byte 1 - Green (low 2 bits are 0)
6418 <1>       ;           byte 2 - Red (low 2 bits are 0)
6419 <1>       ;           byte 3 - pad (or zero byte)
6420 <1>       ;     and 256 colors buffer size must be 256*4 = 1024 bytes
6421 <1>       ;   if BL bit 2 is 0, 6 bit colors will be used directly
6422 <1>       ;       (high two bits of 8 bit color bytes will be 0)
6423 <1>       ;     ((dac color 111111b will be converted to 00111111b))
6424 <1>       ;           byte 0 - Red (high 2 bits are 0)
6425 <1>       ;           byte 1 - Green (high 2 bits are 0)
6426 <1>       ;           byte 2 - Blue (high 2 bits are 0)
6427 <1>       ;     and 256 colors buffer size must be 256*3 = 768 bytes
6428 <1>       ;
6429 <1>       ;     ECX = User's buffer addr (256*3 = 768 bytes) or
6430 <1>       ;           Color
6431 <1>       ;
6432 <1>       ; Return:
6433 <1>       ;     EAX = buffer size (for BL input = 0,1,4,5)
6434 <1>       ;           or color value (for BL input = 2,3,6,7)
6435 <1>       ;
6436 <1>       ; 10/01/2021
6437 <1>       ; SET/READ FONT DATA
6438 <1>       ;
6439 <1>       ;     BH = 12 = Set/Read Character Font Data
6440 <1>       ;
6441 <1>       ;        BL = 0 : Disable system font overwrite
6442 <1>       ;        BL = 1 : Enable system font overwrite
6443 <1>       ;        BL = 2 : Read system font 8x8
6444 <1>       ;        BL = 3 : Read system font 8x14
6445 <1>       ;        BL = 4 : Read system font 8x16
6446 <1>       ;        BL = 5 : Read user defined font 8x8
6447 <1>       ;        BL = 6 : Read user defined font 8x16
6448 <1>       ;        BL = 7 : Write system font 8x8
6449 <1>       ;        BL = 8 : Write system font 8x14
6450 <1>       ;        BL = 9 : Write system font 8x16
6451 <1>       ;        BL = 10 : Write user defined font 8x8
6452 <1>       ;        BL = 11 : Write user defined font 8x16
6453 <1>       ;
6454 <1>       ;       BL > 11 : invalid (not implemented)
6455 <1>       ;
6456 <1>       ;     For BL = 1 to 11
6457 <1>       ;        ECX = number of characters (<= 256)
6458 <1>       ;        EDX = first character (ascii code in DL)
6459 <1>       ;        ESI = user's buffer address
6460 <1>       ;
6461 <1>       ; Return:
6462 <1>       ;     EAX = number of characters (ecx input)
6463 <1>       ;     EAX = 0 -> error
6464 <1>       ;     (EAX = 256 for BL = 0 and 1 if successful)
6465 <1>       ;
6466 <1>       ; Note: system font overwrite permission will be
6467 <1>       ;       given if [multi_tasking] = 0
6468 <1>       ;     and [u.uid] = 0 (BL = 1) ; 19/01/2021
6469 <1>       ;       and if [ufont] bit 7 is 1 (BL = 7,8,9)
6470 <1>       ;
6471 <1>       ; 18/01/2021
6472 <1>       ; SAVE/RESTORE STANDARD VGA VIDEO STATE
6473 <1>       ;
6474 <1>       ;     BH = 13 = Save/Restore std VGA video state
6475 <1>       ;
6476 <1>       ;        BL = 0 : Save VGA state (without DAC regs)
6477 <1>       ;            Return: EAX = VideoStateID (>0)
6478 <1>       ;                  EAX = 0 (failed!)
6479 <1>       ;             (size: 110 bytes for TRDOS 386 v2.0.3)
6480 <1>       ;        BL = 1 : Restore VGA state (without DAC regs)
6481 <1>       ;            ECX = VideoStateID (to be verified)
6482 <1>       ;              Return: EAX = Restore size (>0)
6483 <1>       ;        BL = 2 : Save VGA state (complete)
6484 <1>       ;            Return: EAX = VideoStateID (>0)
6485 <1>       ;                  EAX = 0 (failed!)
6486 <1>       ;             (size: 882 bytes for TRDOS 386 v2.0.3)
6487 <1>       ;        BL = 3 : Restore VGA state (complete)
6488 <1>       ;            ECX = VideoStateID (to be verified)
6489 <1>       ;              Return: EAX = Restore size (>0)
6490 <1>       ;
6491 <1>       ;     * Above options are for saving
6492 <1>       ;     *       video state to system memory
6493 <1>       ;     *     (location: VBE3VIDEOSTATE, 2048 bytes)
6494 <1>       ;
6495 <1>       ;        BL = 4 : Save VGA state (without DAC regs)
6496 <1>       ;            ECX = buffer address
6497 <1>       ;            Return: EAX = transfer count
6498 <1>       ;             (size: 110 bytes for TRDOS 386 v2.0.3)
6499 <1>       ;            ECX = buffer address
6500 <1>       ;        BL = 5 : Restore VGA state (without DAC regs)
6501 <1>       ;            ECX = buffer address
6502 <1>       ;              Return: EAX = transfer count
6503 <1>       ;        BL = 6 : Save VGA state (complete)
6504 <1>       ;            ECX = buffer address
6505 <1>       ;            Return: EAX = transfer count
6506 <1>       ;             (size: 882 bytes for TRDOS 386 v2.0.3)
6507 <1>       ;        BL = 7 : Restore VGA state (complete)
6508 <1>       ;            ECX = buffer address
6509 <1>       ;              Return: EAX = transfer count
6510 <1>       ;
6511 <1>       ;     * Above options are for saving
6512 <1>       ;     *       video state to user's buffer
6513 <1>       ;     *     (buffer size: 110 bytes or 882 bytes)
6514 <1>       ;
6515 <1>       ;       BL > 7 : invalid (not implemented)
6516 <1>       ;
```

```
                                <1>       ; 18/01/2021
                                <1>       ; SAVE/RESTORE SUPER VGA (VESA VBE 2/3) VIDEO STATE
                                <1>       ;
                                <1>       ;     BH = 14 = Save/Restore SVGA video state
                                <1>       ;
                                <1>       ;        BL = options
                                <1>       ;            bit 0 - Save (0) or Restore (1)
                                <1>       ;            bit 1 - controller hardware state
                                <1>       ;            bit 2 - BIOS data state
                                <1>       ;            bit 3 - DAC state
                                <1>       ;            bit 4 - (extended) Register state
                                <1>       ;            bit 5 - system (0) or user (1) memory
                                <1>       ;            bit 6 - verify without transfer
                                <1>       ;            bit 7 - not used (must be 0)
                                <1>       ;
                                <1>       ;          if bit 0 = 0 and bit 5 = 0
                                <1>       ;            Return:      EAX = VideoStateID (>0)
                                <1>       ;          if bit 0 = 1
                                <1>       ;            ECX = VideoStateID (bit 5 = 0)
                                <1>       ;            Return: EAX = restore (transfer) size
                                <1>       ;          if bit 5 = 1
                                <1>       ;            ECX = Buffer address
                                <1>       ;            Return: EAX = transfer count (size)
                                <1>       ;
                                <1>       ;          ECX = Buffer address or VideoStateID
                                <1>       ;
                                <1>       ;       BL > 127 : invalid (not implemented)
                                <1>       ;
                                <1>       ;  Note: Required buffer size may be > 2048 bytes
                                <1>       ;       (function fails when buff size > 2048 bytes)
                                <1>       ;       proper option must be used
                                <1>       ;
                                <1>       ; 18/01/2021
                                <1>       ; READ VESA EDID (EXTENDED DISPLAY IDENTIFICATION DATA)
                                <1>       ;
                                <1>       ;     BH = 15 = Read VESA EDID for connected monitor
                                <1>       ;              (copy EDID to user)
                                <1>       ;
                                <1>       ;        BL = any
                                <1>       ;
                                <1>       ;     Input:
                                <1>       ;         ECX = user's (EDID) buffer address
                                <1>       ;             (buffer size: 128 bytes)
                                <1>       ;     Output:
                                <1>       ;         EAX = 128 (EDID size)
                                <1>       ;         or EAX = 0 -> Error!
                                <1>       ;            (EDID not ready or buffer addr error)
                                <1>       ;
                                <1>
                                <1>       ; 16/05/2016
0000DCF7 31C0                   <1>       xor   eax, eax
0000DCF9 A3[64030300]           <1>       mov   [u.r0], eax
0000DCFE 20FF                   <1>       and   bh, bh
0000DD00 0F858B020000           <1>       jnz   sysvideo_13 ; 11/07/2016
                                <1>
                                <1>       ;; 21/11/2020 (TRDOS 386 v2.0.3)
                                <1>       ;; tty/text mode transfers are only for video mode 3
                                <1>
                                <1>       ; 22/11/2020
                                <1>       ;cmp  byte [CRT_MODE], 3 ; 80x25 text, 16 colors
                                <1>       ;jne  sysret ; invalid (nothing to do), [u.r0] = 0
                                <1>
                                <1>       ; 23/11/2020
                                <1>       ; bit 7,6,5,4 of BL are reserved and it must be 0
                                <1>       ;             for current 'sysvideo' version
                                <1>       ;test  bl, 0F0h
                                <1>       ;;jnz  sysret ; invalid (undefined) !
                                <1>       ;; 28/01/2021
                                <1>       ;jnz  short sysvideo_1_2 ; invalid (undefined) !
                                <1>       ; 28/01/2021
0000DD06 80FB07                 <1>       cmp   bl, 7
0000DD09 776E                   <1>       ja    short sysvideo_1_2 ; invalid (undefined) !
                                <1>
                                <1>       ; Video mode 0, 80*25 text mode, CGA 16 colors
                                <1>       ; [CRT_MODE] = 3
                                <1>       ;mov  bh, bl
                                <1>       ;shr  bh, 2 ; 4..7 -> 1, 8..11 -> 2, 12..15 -> 3
                                <1>       ;; 21/11/2020
                                <1>       ;;and  bh, bh
                                <1>         ;jnz sysvideo_4  ; Display page window transfer etc.
                                <1>
                                <1>       ; 28/01/2021
0000DD0B F6C304                 <1>       test  bl, 4 ; bit 2
0000DD0E 0F85A2000000           <1>       jnz   sysvideo_4 ; Display page window transfer
                                <1>
                                <1>       ; Display page (complete) transfer
0000DD14 80FA07                 <1>       cmp   dl, 7
                                <1>       ;jnz  sysret ; invalid (nothing to do), [u.r0] = 0
0000DD17 760A                   <1>       jna   short sysvideo_0 ; 28/01/2021
0000DD19 FEC2                   <1>       inc   dl ; 0FFh -> 0 ("use current video page")
0000DD1B 755C                   <1>       jnz   short sysvideo_1_2  ; invalid
                                <1>       ; dl = 0 -> use current current page
0000DD1D 8A15[D6800100]         <1>       mov   dl, [ACTIVE_PAGE]
                                <1> sysvideo_0:
                                <1>       ; 28/01/2021
0000DD23 80FB03                 <1>       cmp   bl, 3
0000DD26 7206                   <1>       jb    short sysvideo_0_0
0000DD28 09FF                   <1>       or    edi, edi
0000DD2A 744D                   <1>       jz    short sysvideo_1_2  ; invalid
0000DD2C 89FE                   <1>       mov   esi, edi ; save swap/exchange buffer addr
                                <1>       ; ecx = user buffer for new video page content
                                <1>       ; esi = user (swap) buffer for saving current video page
                                <1> sysvideo_0_0:
0000DD2E BF00800B00             <1>       mov   edi, 0B8000h
                                <1>       ; dl = display page number, destination
```

```
6622 0000DD33 66B80010          <1>       mov   ax, 4096 ; 21/11/2020
6623 0000DD37 20D2              <1>       and   dl, dl
6624 0000DD39 7408              <1>       jz    short sysvideo_1
6625                            <1>       ; 07/02/2021
6626 0000DD3B 88D6              <1>       mov   dh, dl
6627                            <1> sysvideo_0_1:
6628                            <1>       ; page length = 4096 bytes (but page content is 80*25*2 bytes)
6629 0000DD3D 01C7              <1>       add   edi, eax ; 21//11/2020 ([CRT_LEN] = 1000h for mode 3)
6630 0000DD3F FECE              <1>       dec   dh
6631 0000DD41 75FA              <1>       jnz   short sysvideo_0_1
6632                            <1> sysvideo_1:
6633 0000DD43 80E303            <1>       and   bl, 3
6634 0000DD46 7536              <1>       jnz   short sysvideo_2 ; user to system display page transfer
6635                            <1>       ; system to system video page (content) transfer
6636                            <1>       ; cl = display page number, source
6637 0000DD48 80F907            <1>       cmp   cl, 7
6638                            <1>       ;ja   sysret ; invalid (nothing to do), [u.r0] = 0
6639 0000DD4B 760A              <1>       jna   short sysvideo_1_0
6640 0000DD4D FEC1              <1>       inc   cl ; 0FFh -> 0 ("use current video page")
6641 0000DD4F 7528              <1>       jnz   short sysvideo_1_2  ; invalid
6642 0000DD51 8A0D[D6800100]    <1>       mov   cl, [ACTIVE_PAGE]
6643                            <1> sysvideo_1_0:
6644                            <1>       ; 28/01/2021
6645 0000DD57 38D1              <1>       cmp   cl, dl
6646 0000DD59 741E              <1>       je    short sysvideo_1_2  ; same video page !
6647                            <1>
6648                            <1>       ; system to system video/display page transfer (mode 0)
6649                            <1>       ; 21/11/2020
6650                            <1>       ;mov   esi, 0B8000h
6651                            <1>       ;movzx eax, cl
6652                            <1>       ;mov   edx, 4096 ; [CRT_LEN] = 1000h for video mode 3
6653                            <1>       ;mov   ecx, edx
6654                            <1>       ;mul   edx
6655                            <1>       ;add   esi, eax
6656                            <1>       ; 28/01/2021
6657                            <1>       ;movzx esi, cl
6658                            <1>       ;shl   si, 12 ; * 4096
6659                            <1>       ;add   esi, 0B8000h
6660                            <1>
6661                            <1>       ; 28/01/2021
6662 0000DD5B A3[64030300]      <1>       mov   [u.r0], eax ; 4096
6663 0000DD60 BE00800B00        <1>       mov   esi, 0B8000h
6664 0000DD65 08C9              <1>       or    cl, cl
6665 0000DD67 740A              <1>       jz    short sysvideo_1_1
6666                            <1>       ; 07/02/2021
6667 0000DD69 88C8              <1>       mov   al, cl ; display/video page
6668 0000DD6B 30E4              <1>       xor   ah, ah
6669 0000DD6D 66C1E00C          <1>       shl   ax, 12 ; * 4096
6670 0000DD71 01C6              <1>       add   esi, eax
6671                            <1> sysvideo_1_1:
6672                            <1>       ; 21/11/2020
6673                            <1>       ;;mov  ecx, 4096
6674                            <1>       ;mov   ecx, eax ; 4096
6675                            <1>       ;;mov  [u.r0], ecx ; 4096 bytes
6676                            <1>       ; 28/01/2021
6677                            <1>       ;mov   [u.r0], cx
6678 0000DD73 31C9              <1>       xor   ecx, ecx
6679 0000DD75 B504              <1>       mov   ch, 4 ; mov ecx, 1024
6680                            <1>       ;shr   cx, 2 ; / 4
6681 0000DD77 F3A5              <1>       rep   movsd
6682                            <1> sysvideo_1_2:
6683 0000DD79 E952F6FFFF        <1>       jmp   sysret
6684                            <1> sysvideo_2:
6685 0000DD7E 80FB02            <1>       cmp   bl, 2
6686                            <1>       ;ja   sysret; invalid (user to user), [u.r0] = 0
6687                            <1>       ; 28/01/2021
6688 0000DD81 7226              <1>       jb    short sysvideo_3 ; user to system
6689 0000DD83 7404              <1>       je    short sysvideo_2_0 ; system to user
6690                            <1>       ; bl = 3
6691 0000DD85 89CA              <1>       mov   edx, ecx ; save user's buffer addr
6692 0000DD87 89F1              <1>       mov   ecx, esi ; save swap address
6693                            <1> sysvideo_2_0:
6694                            <1>       ; bl = 2 (or bl = 3, stage 1)
6695                            <1>       ; system to user video/display page transfer (mode 0)
6696 0000DD89 89FE              <1>       mov   esi, edi
6697 0000DD8B 89CF              <1>       mov   edi, ecx ; user buffer ; 28/01/2021
6698                            <1>       ; 21/11/2020
6699 0000DD8D 89C1              <1>       mov   ecx, eax ; 4096
6700 0000DD8F E8AE370000        <1>       call  transfer_to_user_buffer ; fast transfer
6701                            <1>       ;jc   sysret ; [u.r0] = 0
6702 0000DD94 72E3              <1>       jc    short sysvideo_1_2 ; 28/01/2021
6703                            <1>       ; 28/01/2021
6704 0000DD96 80FB03            <1>       cmp   bl, 3
6705 0000DD99 7408              <1>       je    short sysvideo_2_2
6706                            <1> sysvideo_2_1:
6707                            <1>       ; 21/11/2020
6708 0000DD9B 890D[64030300]    <1>       mov   [u.r0], ecx
6709                            <1>       ;;mov  [u.r0], cx
6710                            <1>       ;jmp   sysret
6711 0000DDA1 EBD6              <1>       jmp   short sysvideo_1_2
6712                            <1>
6713                            <1> sysvideo_2_2:
6714                            <1>       ; bl = 3 (exchange/swap) complete display page
6715                            <1>       ; esi = video page start address
6716                            <1>       ; edx = user's buffer address
6717                            <1>
6718                            <1>       ;mov   ecx, 4096
6719 0000DDA3 89F7              <1>       mov   edi, esi ; video page start address
6720 0000DDA5 89D6              <1>       mov   esi, edx ; user's (new page) buffer address
6721 0000DDA7 EB04              <1>       jmp   short sysvideo_2_3
6722                            <1> sysvideo_3:
6723                            <1>       ; bl = 1 (or bl = 3, stage 2)
6724                            <1>       ; user to system video/display page transfer (mode 0)
6725 0000DDA9 89CE              <1>       mov   esi, ecx ; user buffer
6726                            <1>       ; edi = video page address
```

```
                                <1>       ; 21/11/2020
0000DDAB 89C1                   <1>       mov    ecx, eax ; 4096
                                <1> sysvideo_2_3:
0000DDAD E8DA370000             <1>       call   transfer_from_user_buffer ; fast transfer
                                <1>       ;jc    sysret ; [u.r0] = 0
0000DDB2 72C5                   <1>       jc     short sysvideo_1_2 ; 28/01/2021
0000DDB4 EBE5                   <1>       jmp    short sysvideo_2_1
                                <1>
                                <1>       ; 21/11/2020
                                <1>       ;mov   [u.r0], ecx
                                <1>       ;;mov  [u.r0], cx
                                <1>       ;;jmp  sysret
                                <1>       ;jmp   short sysvideo_1_2 ; 28/01/2021
                                <1>
                                <1> sysvideo_4:
                                <1>       ; 23/11/2020 (TRDOS 386 v2.0.3)
                                <1>
                                <1>       ; Display page window transfer etc.
0000DDB6 80E303                 <1>       and    bl, 3
0000DDB9 0F85F7000000           <1>         jnz     sysvideo_9 ; user to system or system to user
                                <1>       ; 21/11/2020
                                <1>       ; system to system video/display page window transfer (mode 0)
0000DDBF 81FE00800B00           <1>       cmp    esi, 0B8000h ; source buffer address (system)
0000DDC5 0F8205F6FFFF           <1>       jb     sysret
0000DDCB 81FE00000C00           <1>       cmp    esi, 0B8000h+(4096*8)
0000DDD1 0F83F9F5FFFF           <1>       jnb    sysret
0000DDD7 81FF00800B00           <1>       cmp    edi, 0B8000h ; destination buffer address (system)
0000DDDD 0F82EDF5FFFF           <1>       jb     sysret
0000DDE3 81FF00000C00           <1>       cmp   edi, 0B8000h+(4096*8)
0000DDE9 0F83E1F5FFFF           <1>       jnb    sysret
                                <1>       ;
0000DDEF 51                     <1>       push   ecx ; X1 and Y1 position - top left column, row
0000DDF0 0FB7C1                 <1>       movzx  eax, cx ; top left column
                                <1>       ; 21/11/2020
0000DDF3 C1E910                 <1>       shr    ecx, 16 ; top row
0000DDF6 740E                   <1>       jz     short sysvideo_4_0 ; bypass following code
0000DDF8 52                     <1>       push   edx ; X2 and Y2 position - bottom right column, row
0000DDF9 50                     <1>       push   eax
0000DDFA 66B8A000               <1>       mov    ax, 80*2 ; 80 colums, 160 bytes per row
0000DDFE F7E1                   <1>       mul    ecx
                                <1>              ; eax = offset for start row number
0000DE00 01C6                   <1>       add    esi, eax
0000DE02 01C7                   <1>       add    edi, eax
0000DE04 58                     <1>       pop    eax
0000DE05 5A                     <1>       pop    edx
                                <1> sysvideo_4_0:
0000DE06 66D1E0                 <1>       shl    ax, 1 ; * 2 ; convert start column number to offset
0000DE09 7404                   <1>       jz     short sysvideo_4_1
0000DE0B 01C6                   <1>       add    esi, eax
0000DE0D 01C7                   <1>       add    edi, eax
                                <1>              ; esi = source page window start offset
                                <1>              ; edi = destination page window start offset
                                <1> sysvideo_4_1:
0000DE0F 59                     <1>       pop    ecx
                                <1>       ;mov   eax, 0B8000h+(80*25*2*8)
                                <1>       ; 21/11/2020
0000DE10 B800000C00             <1>       mov    eax, 0B8000h+(4096*8)
0000DE15 39C6                   <1>       cmp    esi, eax
0000DE17 0F83B3F5FFFF           <1>       jnb    sysret ; out of video page
0000DE1D 39C6                   <1>       cmp    esi, eax
0000DE1F 0F83ABF5FFFF           <1>       jnb    sysret  ;out of video page
                                <1>
0000DE25 56                     <1>       push   esi ; ****
0000DE26 57                     <1>       push   edi ; ***
0000DE27 52                     <1>       push   edx ; **
0000DE28 51                     <1>       push   ecx ; *
0000DE29 C1E910                 <1>       shr    ecx, 16 ; top row
0000DE2C C1EA10                 <1>       shr    edx, 16 ; bottom row
                                <1>       ; 21/11/2020
                                <1>       ;cmp   ecx, 24 ; max. 25 rows
0000DE2F 6683F918               <1>       cmp    cx, 24
0000DE33 7778                   <1>       ja     short sysvideo_6 ; invalid, [u.r0] = 0
                                <1>       ;cmp   edx, 24 ; max. 25 rows
0000DE35 6683FA18               <1>       cmp    dx, 24
0000DE39 7772                   <1>       ja     short sysvideo_6 ; invalid, [u.r0] = 0
0000DE3B 28CA                   <1>       sub    dl, cl ; end >= start
0000DE3D 726E                   <1>       jc     short sysvideo_6 ; invalid, [u.r0] = 0
                                <1>       ; 21/11/2020
0000DE3F 89D3                   <1>       mov    ebx, edx ; row count - 1
0000DE41 7415                   <1>       jz     short sysvideo_4_2
0000DE43 50                     <1>       push   eax ; *****
0000DE44 B8A0000000             <1>       mov    eax, 80*2 ; bytes per row
0000DE49 F7E3                   <1>       mul    ebx ; 21/11/2020
                                <1>              ; eax = window end offset
                                <1>              ; (for the last row, before adding column bytes)
0000DE4B 01C6                   <1>       add    esi, eax
0000DE4D 01C7                   <1>       add    edi, eax
0000DE4F 58                     <1>       pop    eax ; ***** ; mode 3 video memory end (0C000h)
0000DE50 39C6                   <1>       cmp    esi, eax
                                <1>       ;ja    short sysvideo_6 ; invalid, [u.r0] = 0
0000DE52 7359                   <1>       jnb    short sysvideo_6 ; 21/11/2020
0000DE54 39C7                   <1>       cmp    edi, eax
                                <1>       ;ja    short sysvideo_6 ; invalid, [u.r0] = 0
0000DE56 7355                   <1>       jnb    short sysvideo_6 ; 21/11/2020
                                <1> sysvideo_4_2:
0000DE58 59                     <1>       pop    ecx ; *
0000DE59 5A                     <1>       pop    edx ; **
0000DE5A 81E1FFFF0000           <1>       and    ecx, 0FFFFh
0000DE60 81E2FFFF0000           <1>       and    edx, 0FFFFh
                                <1>       ; 21/11/2020
                                <1>       ;cmp   ecx, 79 ; max. 80 columns
0000DE66 6683F94F               <1>       cmp    cx, 79
0000DE6A 7743                   <1>       ja     short sysvideo_7 ; invalid, [u.r0] = 0
                                <1>       ;cmp   edx, 79 ; max. 80 columns
0000DE6C 6683FA4F               <1>       cmp    dx, 79
```

```
6832 0000DE70 773D              <1>       ja     short sysvideo_7 ; invalid, [u.r0] = 0
6833 0000DE72 28CA              <1>       sub    dl, cl
6834 0000DE74 7639              <1>       jna    short sysvideo_7 ; invalid, [u.r0] = 0
6835                            <1>       ; 21/11/2020
6836 0000DE76 740E              <1>       jz     short sysvideo_4_3
6837                            <1>       ; edx = column count (width) - 1
6838 0000DE78 D0E2              <1>       shl    dl, 1 ; * 2 ; byte offset (in end row)
6839 0000DE7A 01D6              <1>       add    esi, edx
6840 0000DE7C 01D7              <1>       add    edi, edx
6841                            <1>              ; esi = source page window end offset
6842                            <1>              ; edi = destination page window end offset
6843 0000DE7E 39C6              <1>       cmp    esi, eax ; video memory end
6844                            <1>       ;ja    short sysvideo_7
6845 0000DE80 732D              <1>       jnb    short sysvideo_7 ; 21/11/2020
6846 0000DE82 39C7              <1>       cmp    edi, eax ; video memory end
6847                            <1>       ;ja    short sysvideo_7
6848 0000DE84 7329              <1>       jnb    short sysvideo_7 ; 21/11/2020
6849                            <1> sysvideo_4_3:
6850 0000DE86 5F                <1>       pop    edi ; ***
6851 0000DE87 5E                <1>       pop    esi ; ****
6852 0000DE88 FEC3              <1>       inc    bl ; row count - 1 -> row count
6853 0000DE8A FEC2              <1>       inc    dl ; column count
6854 0000DE8C 88D7              <1>       mov    bh, dl
6855 0000DE8E D0E2              <1>       shl    dl, 1 ; convert column count to byte offset
6856                            <1>       ; 21/11/2020
6857                            <1>       ; esi = source page window start offset
6858                            <1>       ; edi = destination page window start offset
6859 0000DE90 B8A0000000        <1>       mov    eax, 80*2 ; bytes per row
6860                            <1>       ; Note: 160 bytes per row (even if move count < 160)
6861                            <1> sysvideo_5:
6862 0000DE95 88F9              <1>       mov    cl, bh ; move/transfer -word- count per row
6863 0000DE97 0115[64030300]    <1>       add    [u.r0], edx ; transfer count in bytes
6864 0000DE9D F366A5            <1>       rep    movsw
6865 0000DEA0 01C6              <1>       add    esi, eax ; + 160 bytes to next row
6866 0000DEA2 01C7              <1>       add    edi, eax ; + 160 bytes to next row
6867 0000DEA4 FECB              <1>       dec    bl ; remain count of rows
6868 0000DEA6 75ED              <1>       jnz    short sysvideo_5
6869 0000DEA8 E923F5FFFF        <1>       jmp    sysret
6870                            <1>
6871                            <1> sysvideo_6:
6872 0000DEAD 59                <1>       pop    ecx ; *
6873 0000DEAE 5A                <1>       pop    edx ; **
6874                            <1> sysvideo_7:
6875 0000DEAF 5F                <1>       pop    edi ; ***
6876 0000DEB0 5E                <1>       pop    esi ; ****
6877                            <1> sysvideo_8:
6878 0000DEB1 E91AF5FFFF        <1>       jmp    sysret
6879                            <1>
6880                            <1> sysvideo_9:
6881                            <1>       ; user to system or system to user window transfer
6882                            <1>       ; 28/01/2021 (bl = 3 -> swap/exchange)
6883                            <1>       ;cmp   bl, 2
6884                            <1>       ;;ja   sysret ; user to user transfer is invalid
6885                            <1>       ;             ; [u.r0] = 0
6886                            <1>       ;ja    short sysvideo_8 ; 26/12/2020
6887                            <1>
6888                            <1>       ; 28/01/2021
6889 0000DEB6 80FB02            <1>       cmp    bl, 2
6890 0000DEB9 7604              <1>       jna    short sysvideo_9_8
6891                            <1>
6892                            <1>       ; swap/ exchange video memory and user mem windows
6893                            <1>       ; edi = swap address in user's memory space
6894 0000DEBB 21FF              <1>       and    edi, edi
6895 0000DEBD 74F2              <1>       jz     short sysvideo_8 ; invalid ; 28/01/2021
6896                            <1>
6897                            <1> sysvideo_9_8:
6898 0000DEBF 56                <1>       push   esi ; ****
6899 0000DEC0 57                <1>       push   edi ; ***
6900 0000DEC1 52                <1>       push   edx ; **
6901 0000DEC2 51                <1>       push   ecx ; *
6902                            <1>
6903 0000DEC3 C1E910            <1>       shr    ecx, 16 ; top row
6904 0000DEC6 C1EA10            <1>       shr    edx, 16 ; bottom row
6905                            <1>
6906                            <1>       ; 21/11/2020
6907                            <1>       ;cmp   ecx, 24 ; max. 25 rows
6908 0000DEC9 6683F918          <1>       cmp    cx, 24
6909 0000DECD 77DE              <1>       ja     short sysvideo_6 ; invalid, [u.r0] = 0
6910                            <1>       ;cmp   edx, 24 ; max. 25 rows
6911 0000DECF 6683FA18          <1>       cmp    dx, 24
6912 0000DED3 77D8              <1>       ja     short sysvideo_6 ; invalid, [u.r0] = 0
6913 0000DED5 28CA              <1>       sub    dl, cl
6914 0000DED7 72D4              <1>       jc     short sysvideo_6 ; invalid, [u.r0] = 0
6915                            <1>
6916                            <1>       ;mov   ch, cl ; top row
6917                            <1>       ;mov   cl, [ACTIVE_PAGE]
6918                            <1>
6919                            <1>       ;mov   edi, 80*25*2 ; 4000
6920                            <1>       ; 21/11/2020
6921                            <1>       ;mov   edi, 4096 ; [CRT_LEN = 4096 for video mode 3
6922                            <1>       ;shl   edi, cl  ; ! wrong for page 2 to page 7 !
6923                            <1>       ;;add  edi, 0B8000h - 80*25*2
6924                            <1>       ;add   edi, 0B8000h - 1000h ; - 4096
6925                            <1>
6926                            <1>       ; 21/11/2020
6927                            <1>       ;xor   eax, eax
6928                            <1>       ;mov   edi, 0B8000h
6929                            <1>       ;and   cl, cl ; is video page = 0 ?
6930                            <1>       ;jz    short sysvideo_9_1 ; yes
6931                            <1>       ; eax = 0
6932                            <1>
6933                            <1> ;sysvideo_9_0:
6934                            <1>       ;add   ax, 4096
6935                            <1>       ;dec   cl
6936                            <1>       ;jnz   short sysvideo_9_0
```

```
                                <1>       ;add   edi, eax
                                <1>       ;      ; edi = video page start address
                                <1>
                                <1>       ; 21/11/2020
0000DED9 BF00800B00             <1>       mov    edi, 0B8000h
0000DEDE 803D[D6800100]00       <1>       cmp    byte [ACTIVE_PAGE], 0
0000DEE5 760C                   <1>       jna    short sysvideo_9_1
                                <1> stsvideo_9_0:
0000DEE7 B010                   <1>       mov    al, 16 ; 4096/256
0000DEE9 F625[D6800100]         <1>       mul    byte [ACTIVE_PAGE]
0000DEEF 86E0                   <1>       xchg   ah, al ; * 256
0000DEF1 01C7                   <1>       add    edi, eax
                                <1>              ; edi = video page start address
                                <1> sysvideo_9_1:
                                <1>       ; bl = transfer direction
                                <1>       ;     (1 = from user, 2 = to user)
                                <1>       ;     (3 = swap) ; 28/01/2021
0000DEF3 88D7                   <1>       mov    bh, dl ; row count - 1
                                <1>       ;mov   dl, ch ; top row
                                <1>       ; 21/11/2020
0000DEF5 08C9                   <1>       or     cl, cl ; top row number
0000DEF7 7408                   <1>       jz     short sysvideo_9_2
                                <1>
                                <1>       ;mov   eax, 80*2
0000DEF9 66B8A000               <1>       mov    ax, 80*2 ; 160, bytes per row
0000DEFD F7E1                   <1>       mul    ecx ; 22/11/2020
0000DEFF 01C7                   <1>       add    edi, eax
                                <1>              ; edi = window start address for top row
                                <1> sysvideo_9_2:
0000DF01 59                     <1>       pop    ecx ; *
0000DF02 5A                     <1>       pop    edx ; **
0000DF03 81E1FFFF0000           <1>       and    ecx, 0FFFFh
0000DF09 81E2FFFF0000           <1>       and    edx, 0FFFFh
                                <1>       ; 21/11/2020
                                <1>       ;cmp   ecx, 79 ; max. 80 columns
0000DF0F 6683F94F               <1>       cmp    cx, 79
0000DF13 779A                   <1>       ja     short sysvideo_7 ; invalid, [u.r0] = 0
                                <1>       ;cmp   edx, 79 ; max. 80 columns
0000DF15 6683FA4F               <1>       cmp    dx, 79
0000DF19 7794                   <1>       ja     short sysvideo_7 ; invalid, [u.r0] = 0
                                <1>
0000DF1B 28CA                   <1>       sub    dl, cl
0000DF1D 7290                   <1>       jc     short sysvideo_7 ; invalid, [u.r0] = 0
                                <1>
0000DF1F 08C9                   <1>       or     cl, cl ; left column
0000DF21 7404                   <1>       jz     short sysvideo_9_3 ; 0
                                <1>
                                <1>       ; 21/11/2020
0000DF23 D0E1                   <1>       shl    cl, 1  ; column * 2
0000DF25 01CF                   <1>       add    edi, ecx
                                <1>              ; edi = window start addr for top left column
                                <1> sysvideo_9_3:
0000DF27 88D1                   <1>       mov    cl, dl
0000DF29 FEC1                   <1>       inc    cl ; column count
0000DF2B D0E1                   <1>       shl    cl, 1 ; column count * 2
                                <1>              ; ecx = transfer count per row
                                <1>
0000DF2D 58                     <1>       pop    eax ; *** (swap address)
0000DF2E 5E                     <1>       pop    esi ; ****
                                <1>
0000DF2F FEC7                   <1>       inc    bh  ; row count
                                <1>
                                <1>       ;mov   edx, 80*2
0000DF31 B2A0                   <1>       mov    dl, 80*2  ; bytes per row
                                <1>       ;
                                <1>       ;cmp   bl, 1 ; transfer direction
                                <1>       ;ja    short sysvideo_11 ; system to user transfer
                                <1>       ; 28/01/2021
0000DF33 F6C301                 <1>       test   bl, 1
0000DF36 7439                   <1>       jz     short sysvideo_11 ; system to user transfer
                                <1>
                                <1>       ; user to system video/display page window transfer (mode 0)
0000DF38 21C0                   <1>       and    eax, eax ; swap address
0000DF3A 741B                   <1>       jz     short sysvideo_10 ; no window swap
                                <1> sysvideo_9_7: ; 28/01/2021
                                <1>       ; save previous window content in user's buffer (swap address)
0000DF3C 56                     <1>       push   esi ; user buffer
0000DF3D 57                     <1>       push   edi ; beginning address of the window
                                <1>       ; 21/11/2020
0000DF3E 53                     <1>       push   ebx ; save bh
0000DF3F 89FE                   <1>       mov    esi, edi
0000DF41 89C7                   <1>       mov    edi, eax
                                <1> sysvideo_9_4:
0000DF43 E8FA350000             <1>       call   transfer_to_user_buffer ; fast transfer
0000DF48 7208                   <1>       jc     short sysvideo_9_5
                                <1>       ; ecx = actual transfer count (must be same with input)
0000DF4A 01D6                   <1>       add    esi, edx ; next row address of (video page) window
0000DF4C 01CF                   <1>       add    edi, ecx ; next row address of user's window
                                <1>              ; Note: ecx may be less than row length of video page
                                <1>              ; user's window uses offset according to window width
0000DF4E FECF                   <1>       dec    bh
0000DF50 75F1                   <1>       jnz    short sysvideo_9_4 ; repeat for next row
                                <1> sysvideo_9_5:
0000DF52 5B                     <1>       pop    ebx ; restore bh
0000DF53 5F                     <1>       pop    edi
0000DF54 5E                     <1>       pop    esi
                                <1>       ;jnc   short sysvideo_10
0000DF55 7215                   <1>       jc     short sysvideo_9_6 ; 28/01/2021
                                <1> ;sysvideo_9_6:
                                <1> ;     jmp    sysret ; [u.r0] = 0
                                <1>
                                <1> sysvideo_10:
                                <1>       ; user to system video/display page window transfer (mode 0)
                                <1>       ; esi =      user buffer
0000DF57 E830360000             <1>       call   transfer_from_user_buffer ; fast transfer
```

```
7042                                    <1>       ;jc    sysret
7043 0000DF5C 720E                      <1>       jc     short sysvideo_9_6 ; 28/01/2021
7044                                    <1>       ; ecx = actual transfer count (must be same with input)
7045 0000DF5E 010D[64030300]            <1>       add    [u.r0], ecx ; actual transfer count
7046 0000DF64 01D7                      <1>       add    edi, edx ; next row address of (video page) window
7047 0000DF66 01CE                      <1>       add    esi, ecx ; next row address of user's window
7048                                    <1>              ; Note: ecx may be less than row length of video page
7049                                    <1>              ; user's window uses offset according to window width
7050 0000DF68 FECF                      <1>       dec    bh
7051 0000DF6A 75EB                      <1>       jnz    short sysvideo_10 ; repeat for next row
7052                                    <1>       ;jmp   sysret
7053                                    <1> sysvideo_9_6:
7054 0000DF6C E95FF4FFFF                <1>       jmp    sysret
7055                                    <1>
7056                                    <1> sysvideo_11:
7057                                    <1>       ; system to user video/display page window transfer (mode 0)
7058 0000DF71 87FE                      <1>       xchg   edi, esi
7059                                    <1> sysvideo_12:
7060                                    <1>       ; esi = beginning addr of the (screen, video page) window
7061                                    <1>       ; edi =      user's buffer
7062 0000DF73 E8CA350000                <1>       call   transfer_to_user_buffer ; fast transfer
7063 0000DF78 0F8252F4FFFF              <1>       jc     sysret
7064                                    <1>       ; ecx = actual transfer count (must be same with input)
7065 0000DF7E 010D[64030300]            <1>       add    [u.r0], ecx
7066 0000DF84 01D6                      <1>       add    esi, edx ; next row (edx = 160)
7067 0000DF86 01CF                      <1>       add    edi, ecx ; next row of the user's window
7068                                    <1>                       ; (ecx <= 160)
7069 0000DF88 FECF                      <1>       dec    bh
7070 0000DF8A 75E7                      <1>       jnz    short sysvideo_12
7071                                    <1> sysvideo_12_0:
7072 0000DF8C E93FF4FFFF                <1>       jmp    sysret
7073                                    <1>
7074                                    <1> sysvideo_13:
7075                                    <1>       ; 28/12/2020
7076 0000DF91 80FF01                    <1>       cmp    bh, 1
7077 0000DF94 7753                      <1>       ja     short sysvideo_15 ; 23/11/2020
7078                                    <1>
7079                                    <1>       ; 25/02/2021
7080                                    <1>       ; 12/02/2021
7081                                    <1>       ; 29/01/2021, 31/01/2021
7082                                    <1>       ; 23/11/2020 (TRDOS 386 v2.0.3)
7083                                    <1>       ; (major modification, from mode 13h to all VGA modes,
7084                                    <1>       ;  except super VGA modes and liner frame buffer method)
7085                                    <1>
7086                                    <1>       ; BH = 1 = VGA Graphics mode (0A0000h) data transfers
7087                                    <1>
7088                                    <1>       ; 29/01/2021
7089 0000DF96 66B84001                  <1>       mov    ax, 320      ; 320 pixels
7090 0000DF9A F6C380                    <1>       test   bl, 80h      ; bit 7 (screen width, 640 pixels)
7091 0000DF9D 7408                      <1>       jz     short sysvideo_13_0
7092 0000DF9F 66D1E0                    <1>       shl    ax, 1  ; 640 pixels
7093                                    <1>       ;
7094 0000DFA2 80E37F                    <1>       and    bl, 7Fh
7095 0000DFA5 7405                      <1>       jz     short sysvideo_14
7096                                    <1> sysvideo_13_0:
7097                                    <1>       ; 29/01/2021
7098 0000DFA7 80FB41                    <1>       cmp    bl, 41h
7099 0000DFAA 77E0                      <1>       ja     short sysvideo_12_0 ; invalid (unknown) sub function
7100                                    <1> sysvideo_14:
7101 0000DFAC 66A3[6E120300]            <1>       mov    [v_width], ax; save screen width
7102 0000DFB2 C705[72120300]0000-       <1>       mov    dword [v_mem], 0A0000h  ; save video memory address
7102 0000DFBA 0A00                      <1>
7103 0000DFBC C705[76120300]0000-       <1>       mov    dword [v_siz], 65536      ; save video memory size
7103 0000DFC4 0100                      <1>
7104 0000DFC6 C705[7E120300]0000-       <1>       mov    dword [v_end], 0B0000h    ; save end of video page
7104 0000DFCE 0B00                      <1>
7105 0000DFD0 B708                      <1>       mov    bh, 8
7106 0000DFD2 883D[71120300]            <1>       mov    [v_bpp], bh ; 8     ; bits per pixel (256 colors)
7107 0000DFD8 881D[70120300]            <1>       mov    [v_ops], bl  ; VGA data transfer options
7108                                    <1>       ;mov   [maskbuff], edi ; 25/02/2021
7109 0000DFDE 893D[82120300]            <1>       mov    [maskcolor], edi ; 25/02/2021
7110                                    <1>                     ; save mask color or bitmask buffer address
7111 0000DFE4 E9BD000000                <1>       jmp    sysvideo_15_7
7112                                    <1>
7113                                    <1> sysvideo_15:
7114                                    <1>       ; 28/12/2020
7115 0000DFE9 80FF02                    <1>       cmp    bh, 2
7116 0000DFEC 0F87F01E0000              <1>       ja     sysvideo_16
7117                                    <1>
7118                                    <1>       ; 25/02/2021
7119                                    <1>       ; 12/02/2021
7120                                    <1>       ; 30/01/2021, 31/01/2021
7121                                    <1>       ; 01/01/2021, 29/01/2021
7122                                    <1>       ; 26/12/2020, 27/12/2020
7123                                    <1>       ; 25/12/2020 (TRDOS 386 v2.0.3)
7124                                    <1>       ;
7125                                    <1>       ; BH = 2 = SVGA (VESA VBE) Graphics mode (LFB) data transfers
7126                                    <1>
7127                                    <1>       ; 25/12/2020
7128                                    <1>       ; resolution table entry will be saved into EBP register
7129                                    <1>
7130 0000DFF2 803D[4E090000]02          <1>       cmp    byte [vbe3], 2 ; VESA VBE 3 video bios
7131                                    <1>                       ; or BOCHS/QEMU/VIRTUALBOX emu video bios
7132 0000DFF9 724E                      <1>       jb     short sysvideo_15_4 ; no, nothing to do !
7133 0000DFFB 770B                      <1>       ja     short sysvideo_15_0 ; yes
7134                                    <1>
7135                                    <1>       ; Only Bochs/Plex86 (emu) vbe2 video bios is usable in pmid
7136                                    <1>       ; (if [vbe3] = 2)
7137 0000DFFD A0[4F090000]              <1>       mov    al, [vbe2bios] ; Bochs vbios sign is from C0h to C5h
7138 0000E002 24F0                      <1>       and    al, 0F0h
7139 0000E004 3CC0                      <1>       cmp    al, 0C0h
7140 0000E006 7541                      <1>       jne    short sysvideo_15_4 ; unknown (vbe2) video bios
7141                                    <1> sysvideo_15_0:
7142                                    <1>       ; 29/01/2021
7143 0000E008 80FB41                    <1>       cmp    bl, 41h
```

```
7144 0000E00B 773C              <1>       ja     short sysvideo_15_4 ; invalid (unknown) sub function
7145                            <1>       ; 29/01/2021
7146 0000E00D 881D[70120300]    <1>       mov    [v_ops], bl  ; SVGA data transfer options
7147                            <1>
7148 0000E013 89D8              <1>       mov    eax, ebx ; hw of ebx is vesa vbe video mode
7149 0000E015 C1E810            <1>       shr    eax, 16 ; ax = vesa vbe video mode
7150 0000E018 7513              <1>       jnz    short sysvideo_15_2
7151                            <1>       ; ax = 0
7152                            <1>
7153                            <1>       ; check & use current video mode
7154 0000E01A 803D[BA6A0000]FF  <1>       cmp    byte [CRT_MODE], 0FFh ; extended (SVGA) mode ?
7155 0000E021 7526              <1>       jne    short sysvideo_15_4 ; no
7156                            <1> sysvideo_15_1:
7157                            <1>       ; use current vbe (svga) video mode
7158 0000E023 66A1[06120300]    <1>       mov    ax, [video_mode] ; extended (SVGA, VESA VBE) mode
7159 0000E029 6625FF01          <1>       and    ax, 1FFh ; vesa vbe video mode: 1XXh
7160                            <1> sysvideo_15_2:
7161                            <1>       ; 29/01/2021
7162                            <1>       ;mov   [maskbuff], edi ; 25/02/2021
7163 0000E02D 893D[82120300]    <1>       mov    [maskcolor], edi ; 25/02/2021
7164                            <1>                     ; save mask color or bitmask buffer address
7165 0000E033 BD[466E0000]      <1>       mov    ebp, b_vbe_modes ; vbe mode table (in 'vidata.s')
7166                            <1> sysvideo_15_3:
7167 0000E038 663B4500          <1>       cmp    ax, [ebp]
7168 0000E03C 7410              <1>       je     short sysvideo_15_5
7169 0000E03E 83C508            <1>       add    ebp, 8 ; vbe mode table entry size
7170 0000E041 81FD[066F0000]    <1>       cmp    ebp, end_of_b_vbe_modes
7171 0000E047 72EF              <1>       jb     short sysvideo_15_3
7172                            <1> sysvideo_15_4:
7173                            <1>       ; desired video mode is not a valid (implemented)
7174                            <1>       ;       extended (VESA VBE, SVGA) video mode
7175                            <1>       ;
7176                            <1>       ; nothing to do !
7177                            <1>
7178                            <1>       ; [u.r0] = 0  ; return value of EAX
7179 0000E049 E982F3FFFF        <1>       jmp    sysret
7180                            <1>
7181                            <1> sysvideo_15_5:
7182                            <1>       ; get LFB address
7183 0000E04E A1[14120300]      <1>       mov    eax, [LFB_ADDR] ; [LFB_Info+LFBINFO.LFB_addr]
7184 0000E053 09C0              <1>       or     eax, eax
7185 0000E055 7509              <1>       jnz    short sysvideo_15_6
7186 0000E057 66A1[D90E0000]    <1>       mov    ax, [def_LFB_addr] ; default LFB addr
7187                            <1>                               ; (for vbe mode 118h)
7188 0000E05D C1E010            <1>       shl    eax, 16
7189                            <1>       ; 27/12/2020
7190                            <1>       ;jz    short sysvideo_15_4
7191                            <1> sysvideo_15_6:
7192                            <1>       ; 29/01/2021
7193 0000E060 A3[72120300]      <1>       mov    [v_mem], eax ; save video memory address
7194                            <1>
7195                            <1>       ; 27/12/2020
7196                            <1>       ; 26/12/2020
7197 0000E065 8B4502            <1>       mov    eax, [ebp+2] ; width, height
7198                            <1>       ; 29/01/2021
7199 0000E068 66A3[6E120300]    <1>       mov    [v_width], ax ; save screen width
7200                            <1>       ; 28/12/2020
7201 0000E06E 8A7D06            <1>       mov    bh, [ebp+6] ; bpp
7202                            <1>       ; 28/02/2021
7203                            <1>       ; check default truecolor bpp value and use
7204                            <1>       ; 32bpp instead of 24bpp if the default value
7205                            <1>       ; has been set to 32bpp.
7206 0000E071 80FF18            <1>       cmp    bh, 24
7207 0000E074 750B              <1>       jne    short sysvideo_15_16
7208 0000E076 803D[6F7D0100]20  <1>       cmp    byte [truecolor], 32
7209                            <1>              ; Default truecolor bpp value,
7210                            <1>              ; it is 32 for VBE3 video bios
7211                            <1>              ; (it can be set to 32 or 24)
7212 0000E07D 7502              <1>       jne    short sysvideo_15_16 ; not VBE3 !
7213                            <1>                               ; or it is set to 24
7214 0000E07F B720              <1>       mov    bh, 32
7215                            <1>       ; 28/02/2021
7216                            <1> sysvideo_15_16:
7217                            <1>       ; 29/01/2021
7218 0000E081 883D[71120300]    <1>       mov    [v_bpp], bh ; bits per pixel
7219                            <1>
7220 0000E087 52                <1>       push   edx ; *
7221 0000E088 0FB7D0            <1>       movzx  edx, ax  ; width
7222 0000E08B C1E810            <1>       shr    eax, 16  ; height
7223 0000E08E F7E2              <1>       mul    edx
7224                            <1>       ; eax = linear frame buffer size (pixels)
7225                            <1>       ; 29/01/2021
7226 0000E090 A3[76120300]      <1>       mov    [v_siz], eax ; save video page size
7227 0000E095 E8FD000000        <1>       call   pixels_to_byte_count
7228 0000E09A 0305[72120300]    <1>       add    eax, [v_mem]
7229 0000E0A0 A3[7E120300]      <1>       mov    [v_end], eax ; save end of video page
7230 0000E0A5 5A                <1>       pop    edx ; *
7231                            <1>
7232                            <1>       ; bh = bits per pixel
7233                            <1>       ; (bh will not be used after here, 29/01/2021)
7234                            <1>
7235                            <1>       ; bl = pixel operations & options
7236                            <1>       ; ecx, edx, esi, edi input parameters
7237                            <1>       ; [maskcolor] = edi input ; 25/02/2021
7238                            <1>
7239                            <1> sysvideo_15_7:
7240                            <1>       ; 29/01/2021
7241                            <1>       ;test  byte [v_ops], 40h  ; system to user ?
7242 0000E0A6 F6C340            <1>       test   bl, 40h
7243 0000E0A9 7517              <1>       jnz    short sysvideo_15_9
7244                            <1>
7245 0000E0AB 31C0              <1>       xor    eax, eax
7246 0000E0AD 88D8              <1>       mov    al, bl
7247 0000E0AF BB[D5E10000]      <1>       mov    ebx, pixel_ops
7248 0000E0B4 240F              <1>       and    al, 0Fh ; isolate 16 pixel operations
```

```
7249 0000E0B6 C0E002              <1>       shl    al, 2 ; * 4 for dword table pointers
7250 0000E0B9 01C3                <1>       add    ebx, eax
7251                              <1>
7252                              <1>       ; ebx = subroutine address
7253                              <1>
7254                              <1>       ; ecx, edx, esi, edi input parameters
7255                              <1>       ; [maskbuff] = edi input
7256                              <1>       ; [maskcolor] = edi input ; 25/02/2021
7257                              <1>
7258 0000E0BB FF13                <1>       call   [ebx]
7259                              <1> sysvideo_15_8:
7260 0000E0BD E90EF3FFFF          <1>       jmp    sysret
7261                              <1>
7262                              <1> sysvideo_15_9:
7263                              <1>       ; system to user display page or window copy
7264                              <1>       ;test  byte [v_ops], 1 ; window copy ?
7265 0000E0C2 F6C301              <1>       test   bl, 1
7266 0000E0C5 7521                <1>       jnz    short sysvideo_15_10
7267                              <1>
7268                              <1>       ; display page (full screen copy)
7269 0000E0C7 8B35[72120300]      <1>       mov    esi, [v_mem] ; LFB start address
7270 0000E0CD A1[76120300]        <1>       mov    eax, [v_siz]
7271 0000E0D2 E8C0000000          <1>       call   pixels_to_byte_count
7272 0000E0D7 89C1                <1>       mov    ecx, eax ; transfer count in bytes
7273                              <1>       ;edi = user's buffer address
7274 0000E0D9 E864340000          <1>       call   transfer_to_user_buffer
7275 0000E0DE 72DD                <1>       jc     short sysvideo_15_8
7276 0000E0E0 890D[64030300]      <1>       mov    [u.r0], ecx
7277 0000E0E6 EBD5                <1>       jmp    short sysvideo_15_8
7278                              <1>
7279                              <1> sysvideo_15_10:
7280 0000E0E8 E820000000          <1>       call   sysvideo_15_12 ; window preparations
7281 0000E0ED 72CE                <1>       jc     short sysvideo_15_8
7282                              <1>
7283 0000E0EF 8B35[7A120300]      <1>       mov    esi, [v_str]
7284                              <1> sysvideo_15_11:
7285                              <1>       ; esi = window's current row address (video mem)
7286                              <1>       ; edi = current row (virtual) addr in user's buff
7287                              <1>       ; ecx = transfer count per row
7288 0000E0F5 E848340000          <1>       call   transfer_to_user_buffer
7289 0000E0FA 72C1                <1>       jc     short sysvideo_15_8
7290 0000E0FC 010D[64030300]      <1>       add    [u.r0], ecx
7291 0000E102 4B                  <1>       dec    ebx
7292 0000E103 74B8                <1>       jz     short sysvideo_15_8 ; ok.
7293                              <1>       ; next row
7294 0000E105 01CF                <1>       add    edi, ecx ; next row in user's buffer
7295 0000E107 01D6                <1>       add    esi, edx ; next row of window (system)
7296 0000E109 EBEA                <1>       jmp    short sysvideo_15_11
7297                              <1>
7298                              <1> sysvideo_15_14:
7299 0000E10B F9                  <1>       stc     ; error !
7300                              <1> sysvideo_15_15:
7301 0000E10C C3                  <1>       retn
7302                              <1>
7303                              <1> sysvideo_15_12:
7304                              <1>       ; 30/01/2021
7305                              <1>       ; 29/01/2021
7306                              <1>       ; Window address preparations for window copy
7307 0000E10D 6621D2              <1>       and    dx, dx
7308 0000E110 74F9                <1>       jz     short sysvideo_15_14 ; invalid (zero columns)
7309                              <1>       ;test  edx, 0FFFF0000h
7310                              <1>       ;jz    short sysvideo_15_14 ; invalid (zero rows)
7311 0000E112 81FA00000100        <1>       cmp    edx, 65536
7312 0000E118 72F2                <1>       jb     short sysvideo_15_15 ; invalid (zero rows)
7313 0000E11A 89C8                <1>       mov    eax, ecx ; start position (row, column)
7314 0000E11C E899000000          <1>       call   calc_pixel_offset
7315 0000E121 3B05[76120300]      <1>       cmp    eax, [v_siz]
7316 0000E127 73E2                <1>       jnb    short sysvideo_15_14 ; out of display page
7317                              <1>                                   ; nothing to do
7318 0000E129 E869000000          <1>       call   pixels_to_byte_count
7319 0000E12E 0305[72120300]      <1>       add    eax, [v_mem]
7320 0000E134 A3[7A120300]        <1>       mov    [v_str], eax ; window start address
7321                              <1>                             ; (addr of top left corner)
7322                              <1>       ; check column limit
7323 0000E139 89C8                <1>       mov    eax, ecx
7324 0000E13B 6601D0              <1>       add    ax, dx  ; add columns to start column
7325 0000E13E 72CC                <1>       jc     short sysvideo_15_15 ; cf = 1
7326 0000E140 663B05[6E120300]    <1>       cmp    ax, [v_width]
7327 0000E147 77C2                <1>       ja     short sysvideo_15_14
7328                              <1>
7329 0000E149 89D0                <1>       mov    eax, edx ; size
7330 0000E14B 2D00000100          <1>       sub    eax, 65536 ; row count -> 0 based row #
7331 0000E150 E865000000          <1>       call   calc_pixel_offset
7332 0000E155 3B05[76120300]      <1>       cmp    eax, [v_siz] ; video (display) page size
7333 0000E15B 77AE                <1>       ja     short sysvideo_15_14 ; out of display page
7334                              <1>                                   ; nothing to do
7335 0000E15D E835000000          <1>       call   pixels_to_byte_count
7336 0000E162 0305[7A120300]      <1>       add    eax, [v_str] ; window start address
7337 0000E168 3B05[7E120300]      <1>       cmp    eax, [v_end] ; window end address (+1)
7338                              <1>                           ; (addr of bottom right corner +1)
7339 0000E16E 779B                <1>       ja     short sysvideo_15_14 ; out of display page
7340                              <1>                                   ; nothing to do
7341 0000E170 89D3                <1>       mov    ebx, edx
7342 0000E172 C1EB10              <1>       shr    ebx, 16
7343                              <1>       ; ebx = row count
7344 0000E175 81E2FFFF0000        <1>       and    edx, 0FFFFh
7345                              <1>       ; edx = transfer count per row (from user's buffer)
7346                              <1>       ;      (in pixels, window width)
7347 0000E17B 89D0                <1>       mov    eax, edx
7348 0000E17D A3[86120300]        <1>       mov    [pixcount], eax ; 27/02/2021
7349 0000E182 E810000000          <1>       call   pixels_to_byte_count
7350 0000E187 89C1                <1>       mov    ecx, eax
7351                              <1>       ; ecx = transfer count per row (from user's buffer)
7352                              <1>       ;      (in bytes, window width)
7353 0000E189 66A1[6E120300]      <1>       mov    ax, [v_width]
```

```
7354 0000E18F E803000000          <1>       call   pixels_to_byte_count
7355 0000E194 89C2                <1>       mov    edx, eax
7356                              <1>       ; edx = byte count per row
7357 0000E196 C3                  <1>       retn ; cf = 0
7358                              <1>
7359                              <1> pixels_to_byte_count:
7360                              <1>       ; 29/01/2021
7361                              <1>       ; INPUT:
7362                              <1>       ;   eax = pixel count
7363                              <1>       ; OUTPUT:
7364                              <1>       ;   eax = byte count
7365                              <1>       ;
7366 0000E197 803D[71120300]08    <1>       cmp    byte [v_bpp], 8
7367 0000E19E 7619                <1>       jna    short pixtobc_3 ; 8 bit colors
7368 0000E1A0 803D[71120300]18    <1>       cmp    byte [v_bpp], 24
7369 0000E1A7 720A                <1>       jb     short pixtobc_1 ; 16 bit colors
7370 0000E1A9 770B                <1>       ja     short pixtobc_2 ; 32 bit colors
7371                              <1>       ; 24 bit pixels
7372                              <1>       ; eax = eax * 3
7373                              <1>       ;push  edx
7374                              <1>       ;mov   edx, eax
7375                              <1>       ;shl   eax, 1
7376                              <1>       ;add   eax, edx
7377                              <1>       ;pop   edx
7378 0000E1AB 50                  <1>       push   eax
7379 0000E1AC D1E0                <1>       shl    eax, 1
7380 0000E1AE 010424              <1>       add    [esp], eax
7381 0000E1B1 58                  <1>       pop    eax
7382 0000E1B2 C3                  <1>       retn
7383                              <1> pixtobc_1:
7384                              <1>       ; 32 bit pixels
7385                              <1>       ; eax = eax * 2
7386 0000E1B3 D1E0                <1>       shl    eax, 1
7387 0000E1B5 C3                  <1>       retn
7388                              <1> pixtobc_2:
7389                              <1>       ; 16 bit pixels
7390                              <1>       ; eax = eax * 4
7391 0000E1B6 C1E002              <1>       shl    eax, 2
7392                              <1> pixtobc_3:
7393 0000E1B9 C3                  <1>       retn
7394                              <1>
7395                              <1> calc_pixel_offset:
7396                              <1>       ; 29/01/2021
7397                              <1>       ; INPUT:
7398                              <1>       ;   eax = pixel position (row, column)
7399                              <1>       ; OUTPUT:
7400                              <1>       ;   eax = pixel offset (linear address)
7401                              <1>       ;
7402 0000E1BA 52                  <1>       push   edx
7403 0000E1BB 50                  <1>       push   eax
7404 0000E1BC C1E810              <1>       shr    eax, 16
7405 0000E1BF 7409                <1>       jz     short cpixo_0
7406                              <1>       ; eax = row
7407 0000E1C1 0FB715[6E120300]    <1>       movzx  edx, word [v_width]
7408 0000E1C8 F7E2                <1>       mul    edx
7409                              <1> cpixo_0:
7410                              <1>       ; eax = row * screen width
7411 0000E1CA 5A                  <1>       pop    edx
7412 0000E1CB 81E2FFFF0000        <1>       and    edx, 0FFFFh
7413                              <1>       ; edx = column
7414 0000E1D1 01D0                <1>       add    eax, edx
7415                              <1>       ; eax = (row * screen width) + column
7416 0000E1D3 5A                  <1>       pop    edx
7417 0000E1D4 C3                  <1>       retn
7418                              <1>
7419                              <1>       ; 02/02/2021
7420                              <1>       ; 29/01/2021
7421                              <1> pixel_ops:
7422 0000E1D5 [15E20000]          <1>       dd     pix_op_cpy ; copy pixels (user to system)
7423 0000E1D9 [60E70000]          <1>       dd     pix_op_new ; change (new, fill) color
7424 0000E1DD [7DE20000]          <1>       dd     pix_op_add ; add color (up to 0FFh)
7425 0000E1E1 [2FE30000]          <1>       dd     pix_op_sub ; sub color (down to 0)
7426 0000E1E5 [4AE50000]          <1>       dd     pix_op_orc ; or color
7427 0000E1E9 [FCE50000]          <1>       dd     pix_op_and ; and color
7428 0000E1ED [AEE60000]          <1>       dd     pix_op_xor ; xor color
7429 0000E1F1 [24E80000]          <1>       dd     pix_op_not ; not color
7430 0000E1F5 [CEE80000]          <1>       dd     pix_op_neg ; neg color
7431 0000E1F9 [78E90000]          <1>       dd     pix_op_inc ; inc color
7432 0000E1FD [22EA0000]          <1>       dd     pix_op_dec ; dec color
7433 0000E201 [E1E30000]          <1>       dd     pix_op_mix ; mix color
7434 0000E205 [A8E40000]          <1>       dd     pix_op_rpl ; replace color
7435 0000E209 [CCEA0000]          <1>       dd     pix_op_blk ; copy pixel block(s) (sys)
7436 0000E20D [7BEB0000]          <1>       dd     pix_op_lin ; write line(s)
7437 0000E211 [5EEF0000]          <1>       dd     pix_op_chr ; write character (font)
7438                              <1>
7439                              <1> pix_op_cpy:
7440                              <1>       ; 21/02/2021
7441                              <1>       ; 06/02/2021
7442                              <1>       ; 30/01/2021
7443                              <1>       ; COPY PIXELS
7444                              <1>       ;
7445                              <1>       ; INPUT:
7446                              <1>       ;  If bit 4 of BL or [v_ops] = 1 -window copy-
7447                              <1>       ;  ECX = start position (row, column)
7448                              <1>       ;         (HW = row, CX = column)
7449                              <1>       ;  EDX = size (rows, colums)
7450                              <1>       ;         (HW = rows, DX = columns)
7451                              <1>       ;       (0 -> invalid
7452                              <1>       ;         (1 -> horizontal or vertical line)
7453                              <1>       ;  If bit 4 of BL or [v_ops] = 0 -full screen-
7454                              <1>       ;     ECX and EDX will not be used
7455                              <1>       ;  ESI = user's buffer address
7456                              <1>       ;  [maskcolor] = mask color (to be excluded)
7457                              <1>       ;
7458                              <1>       ; OUTPUT:
```

```
7459                                    <1>       ;      [u.r0] will be > 0 if succesful
7460                                    <1>
7461 0000E215 F605[70120300]10          <1>       test   byte [v_ops], 10h ; display page or window ?
7462 0000E21C 752E                      <1>       jnz    short pix_op_cpy_w ; window
7463                                    <1>
7464 0000E21E 8B3D[72120300]            <1>       mov    edi, [v_mem] ; 21/02/2021
7465                                    <1>
7466                                    <1>       ; Copy user's buffer content do display page
7467                                    <1>       ; (full screen copy)
7468 0000E224 A1[76120300]              <1>       mov    eax, [v_siz] ; video page size
7469 0000E229 E869FFFFFF                <1>       call   pixels_to_byte_count
7470 0000E22E 89C1                      <1>       mov    ecx, eax ; transfer count
7471                                    <1>       ; esi = user's buffer address (virtual)
7472 0000E230 F605[70120300]20          <1>       test   byte [v_ops], 20h ; masked copy ?
7473 0000E237 7405                      <1>       jz     short pix_op_cpy_0 ; no
7474 0000E239 E9720F0000                <1>       jmp    m_pix_op_cpy ; copy pixels except mask color
7475                                    <1> pix_op_cpy_0:
7476                                    <1>       ; esi = user buffer for full screen copy
7477                                    <1>       ; edi = start of video memory
7478                                    <1>       ;      (start of display page)
7479                                    <1>       ; ecx = byte count (display page size in bytes)
7480 0000E23E E849330000                <1>       call   transfer_from_user_buffer
7481 0000E243 7206                      <1>       jc     short pix_op_cpy_1
7482 0000E245 890D[64030300]            <1>       mov    [u.r0], ecx
7483                                    <1> pix_op_cpy_1:
7484 0000E24B C3                        <1>       retn   ; 06/02/2021
7485                                    <1>
7486                                    <1> pix_op_cpy_w:
7487 0000E24C E8BCFEFFFF                <1>       call   sysvideo_15_12 ; window preparations
7488 0000E251 72F8                      <1>       jc     short pix_op_cpy_1
7489                                    <1>       ; ecx = bytes per row (to be applied)
7490                                    <1>       ; edx = screen width in bytes
7491                                    <1>       ; ebx = row count
7492 0000E253 8B3D[7A120300]            <1>       mov    edi, [v_str]
7493 0000E259 F605[70120300]20          <1>       test   byte [v_ops], 20h ; masked copy ?
7494 0000E260 7405                      <1>       jz     short pix_op_cpy_w_0 ; no
7495 0000E262 E90C100000                <1>       jmp    m_pix_op_cpy_w ; window copy except mask color
7496                                    <1> pix_op_cpy_w_0:
7497                                    <1>       ; esi = current row (virtual) addr in user's buff
7498                                    <1>       ; edi = window's current row address (video mem)
7499                                    <1>       ; ecx = transfer count per row
7500 0000E267 E820330000                <1>       call   transfer_from_user_buffer
7501 0000E26C 72DD                      <1>       jc     short pix_op_cpy_1
7502 0000E26E 010D[64030300]            <1>       add    [u.r0], ecx
7503 0000E274 4B                        <1>       dec    ebx
7504 0000E275 74D4                      <1>       jz     short pix_op_cpy_1 ; ok.
7505                                    <1>       ; next row
7506 0000E277 01CE                      <1>       add    esi, ecx ; next row in user's buffer
7507 0000E279 01D7                      <1>       add    edi, edx ; next row of window (system)
7508 0000E27B EBEA                      <1>       jmp    short pix_op_cpy_w_0
7509                                    <1>
7510                                    <1> pix_op_add:
7511                                    <1>       ; 31/01/2021
7512                                    <1>       ; 30/01/2021
7513                                    <1>       ; ADD COLOR
7514                                    <1>       ;
7515                                    <1>       ; INPUT:
7516                                    <1>       ;   CL = color (8 bit, 256 colors)
7517                                    <1>       ;  ECX = color (16 bit and true colors)
7518                                    <1>       ;  EDX = start position (row, column)
7519                                    <1>       ;        (HW = row, DX = column)
7520                                    <1>       ;  ESI = size (rows, colums)
7521                                    <1>       ;        (HW = rows, SI = columns)
7522                                    <1>       ;
7523                                    <1>       ;  [maskcolor] = mask color (to be excluded)
7524                                    <1>       ;
7525                                    <1>       ; OUTPUT:
7526                                    <1>       ;      [u.r0] will be > 0 if succesful
7527                                    <1>
7528 0000E27D F605[70120300]10          <1>       test   byte [v_ops], 10h ; display page or window ?
7529 0000E284 7555                      <1>       jnz    short pix_op_add_w ; window
7530                                    <1>
7531 0000E286 8B3D[72120300]            <1>       mov    edi, [v_mem]
7532 0000E28C 89FE                      <1>       mov    esi, edi
7533                                    <1>       ; ecx = color (CL, CX, ECX)
7534 0000E28E 89C8                      <1>       mov    eax, ecx
7535 0000E290 8B0D[76120300]            <1>       mov    ecx, [v_siz] ; display page pixel count
7536                                    <1>
7537 0000E296 F605[70120300]20          <1>       test   byte [v_ops], 20h ; masked color adding ?
7538 0000E29D 7405                      <1>       jz     short pix_op_add_0 ; no
7539 0000E29F E9CE100000                <1>       jmp    m_pix_op_add ; add color except mask color
7540                                    <1> pix_op_add_0:
7541 0000E2A4 803D[71120300]08          <1>       cmp    byte [v_bpp], 8 ; 8bpp
7542 0000E2AB 7707                      <1>       ja     short pix_op_add_1
7543                                    <1>
7544                                    <1>       ; 256 colors (8bpp)
7545 0000E2AD E84D0A0000                <1>       call   pix_op_add_8
7546 0000E2B2 EB1E                      <1>       jmp    short pix_op_add_4
7547                                    <1>
7548                                    <1> pix_op_add_1:
7549 0000E2B4 803D[71120300]18          <1>       cmp    byte [v_bpp], 24 ; 24bpp
7550 0000E2BB 7710                      <1>       ja     short pix_op_add_3 ; 32bpp
7551 0000E2BD 7207                      <1>       jb     short pix_op_add_2 ; 16bpp
7552                                    <1>
7553                                    <1>       ; 24 bit true colors
7554 0000E2BF E85B0A0000                <1>       call   pix_op_add_24
7555 0000E2C4 EB0C                      <1>       jmp    short pix_op_add_4
7556                                    <1>
7557                                    <1>       ; 65536 colors (16bpp)
7558                                    <1> pix_op_add_2:
7559 0000E2C6 E8420A0000                <1>       call   pix_op_add_16
7560 0000E2CB EB05                      <1>       jmp    short pix_op_add_4
7561                                    <1>
7562                                    <1>       ; 32 bit true colors
7563                                    <1> pix_op_add_3:
```

```
7564 0000E2CD E86D0A0000          <1>       call   pix_op_add_32
7565                              <1> pix_op_add_4:
7566 0000E2D2 29F7                <1>       sub    edi, esi
7567 0000E2D4 893D[64030300]      <1>       mov    [u.r0], edi
7568                              <1> pix_op_add_5:
7569 0000E2DA C3                  <1>       retn
7570                              <1>
7571                              <1> pix_op_add_w:
7572                              <1>       ; 31/01/2021
7573 0000E2DB 51                  <1>       push   ecx ; * ; color
7574 0000E2DC 89D1                <1>       mov    ecx, edx ; win start pos
7575 0000E2DE 89F2                <1>       mov    edx, esi ; size (rows, cols)
7576 0000E2E0 E828FEFFFF          <1>       call   sysvideo_15_12 ; window preparations
7577 0000E2E5 58                  <1>       pop    eax ; * ; color
7578 0000E2E6 72F2                <1>       jc     short pix_op_add_5
7579                              <1>
7580 0000E2E8 F605[70120300]20    <1>       test   byte [v_ops], 20h ; masked color adding ?
7581 0000E2EF 7405                <1>       jz     short pix_op_add_w_0 ; no
7582 0000E2F1 E92A110000          <1>       jmp    m_pix_op_add_w
7583                              <1>                      ; window add color except mask color
7584                              <1> pix_op_add_w_0:
7585                              <1>       ; ecx = bytes per row (to be applied)
7586                              <1>       ; edx = screen width in bytes
7587                              <1>       ; ebx = row count
7588                              <1>       ; eax = color
7589                              <1>
7590 0000E2F6 8B3D[7A120300]      <1>       mov    edi, [v_str]
7591 0000E2FC 803D[71120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
7592 0000E303 7707                <1>       ja     short pix_op_add_w_1
7593                              <1>
7594                              <1>       ; 256 colors (8bpp)
7595 0000E305 BD[FFEC0000]        <1>       mov    ebp, pix_op_add_8
7596 0000E30A EB1E                <1>       jmp    short pix_op_add_w_4
7597                              <1>
7598                              <1> pix_op_add_w_1:
7599 0000E30C 803D[71120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
7600 0000E313 7710                <1>       ja     short pix_op_add_w_3 ; 32bpp
7601 0000E315 7207                <1>       jb     short pix_op_add_w_2 ; 16bpp
7602                              <1>
7603                              <1>       ; 24 bit true colors
7604 0000E317 BD[1FED0000]        <1>       mov    ebp, pix_op_add_24
7605 0000E31C EB0C                <1>       jmp    short pix_op_add_w_4
7606                              <1>
7607                              <1>       ; 65536 colors (16bpp)
7608                              <1> pix_op_add_w_2:
7609 0000E31E BD[0DED0000]        <1>       mov    ebp, pix_op_add_16
7610 0000E323 EB05                <1>       jmp    short pix_op_add_w_4
7611                              <1>
7612                              <1>       ; 32 bit true colors
7613                              <1> pix_op_add_w_3:
7614 0000E325 BD[3FED0000]        <1>       mov    ebp, pix_op_add_32
7615                              <1> pix_op_add_w_4:
7616 0000E32A E95F010000          <1>       jmp    pix_op_add_w_x
7617                              <1>
7618                              <1> pix_op_sub:
7619                              <1>       ; 31/01/2021
7620                              <1>       ; SUB COLOR
7621                              <1>       ;
7622                              <1>       ; INPUT:
7623                              <1>       ;   CL = color (8 bit, 256 colors)
7624                              <1>       ;  ECX = color (16 bit and true colors)
7625                              <1>       ;  EDX = start position (row, column)
7626                              <1>       ;        (HW = row, DX = column)
7627                              <1>       ;  ESI = size (rows, colums)
7628                              <1>       ;        (HW = rows, SI = columns)
7629                              <1>       ;
7630                              <1>       ;  [maskcolor] = mask color (to be excluded)
7631                              <1>       ;
7632                              <1>       ; OUTPUT:
7633                              <1>       ;      [u.r0] will be > 0 if succesful
7634                              <1>
7635 0000E32F F605[70120300]10    <1>       test   byte [v_ops], 10h ; display page or window ?
7636 0000E336 7555                <1>       jnz    short pix_op_sub_w ; window
7637                              <1>
7638 0000E338 8B3D[72120300]      <1>       mov    edi, [v_mem]
7639 0000E33E 89FE                <1>       mov    esi, edi
7640                              <1>       ; ecx = color (CL, CX, ECX)
7641 0000E340 89C8                <1>       mov    eax, ecx
7642 0000E342 8B0D[76120300]      <1>       mov    ecx, [v_siz] ; display page pixel count
7643                              <1>
7644 0000E348 F605[70120300]20    <1>       test   byte [v_ops], 20h ; masked color subtract ?
7645 0000E34F 7405                <1>       jz     short pix_op_sub_0 ; no
7646 0000E351 E9FD100000          <1>       jmp    m_pix_op_sub ; sub color except mask color
7647                              <1> pix_op_sub_0:
7648 0000E356 803D[71120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
7649 0000E35D 7707                <1>       ja     short pix_op_sub_1
7650                              <1>
7651                              <1>       ; 256 colors (8bpp)
7652 0000E35F E8EA090000          <1>       call   pix_op_sub_8
7653 0000E364 EB1E                <1>       jmp    short pix_op_sub_4
7654                              <1>
7655                              <1> pix_op_sub_1:
7656 0000E366 803D[71120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
7657 0000E36D 7710                <1>       ja     short pix_op_sub_3 ; 32bpp
7658 0000E36F 7207                <1>       jb     short pix_op_sub_2 ; 16bpp
7659                              <1>
7660                              <1>       ; 24 bit true colors
7661 0000E371 E8FB090000          <1>       call   pix_op_sub_24
7662 0000E376 EB0C                <1>       jmp    short pix_op_sub_4
7663                              <1>
7664                              <1>       ; 65536 colors (16bpp)
7665                              <1> pix_op_sub_2:
7666 0000E378 E8E1090000          <1>       call   pix_op_sub_16
7667 0000E37D EB05                <1>       jmp    short pix_op_sub_4
7668                              <1>
```

```
7669                                    <1>       ; 32 bit true colors
7670                                    <1> pix_op_sub_3:
7671 0000E37F E8070A0000                <1>       call   pix_op_sub_32
7672                                    <1> pix_op_sub_4:
7673 0000E384 29F7                      <1>       sub    edi, esi
7674 0000E386 893D[64030300]            <1>       mov    [u.r0], edi
7675                                    <1> pix_op_sub_5:
7676 0000E38C C3                        <1>       retn
7677                                    <1>
7678                                    <1> pix_op_sub_w:
7679                                    <1>       ; 31/01/2021
7680 0000E38D 51                        <1>       push   ecx ; * ; color
7681 0000E38E 89D1                      <1>       mov    ecx, edx ; win start pos
7682 0000E390 89F2                      <1>       mov    edx, esi ; size (rows, cols)
7683 0000E392 E876FDFFFF                <1>       call   sysvideo_15_12 ; window preparations
7684 0000E397 58                        <1>       pop    eax ; * ; color
7685 0000E398 72F2                      <1>       jc     short pix_op_sub_5
7686                                    <1>
7687 0000E39A F605[70120300]20          <1>       test   byte [v_ops], 20h ; masked color subtract ?
7688 0000E3A1 7405                      <1>       jz     short pix_op_sub_w_0 ; no
7689 0000E3A3 E94E110000                <1>       jmp    m_pix_op_sub_w
7690                                    <1>                     ; window sub color except mask color
7691                                    <1> pix_op_sub_w_0:
7692                                    <1>       ; ecx = bytes per row (to be applied)
7693                                    <1>       ; edx = screen width in bytes
7694                                    <1>       ; ebx = row count
7695                                    <1>       ; eax = color
7696                                    <1>
7697 0000E3A8 8B3D[7A120300]            <1>       mov    edi, [v_str]
7698 0000E3AE 803D[71120300]08          <1>       cmp    byte [v_bpp], 8 ; 8bpp
7699 0000E3B5 7707                      <1>       ja     short pix_op_sub_w_1
7700                                    <1>
7701                                    <1>       ; 256 colors (8bpp)
7702 0000E3B7 BD[4EED0000]              <1>       mov    ebp, pix_op_sub_8
7703 0000E3BC EB1E                      <1>       jmp    short pix_op_sub_w_4
7704                                    <1>
7705                                    <1> pix_op_sub_w_1:
7706 0000E3BE 803D[71120300]18          <1>       cmp    byte [v_bpp], 24 ; 24bpp
7707 0000E3C5 7710                      <1>       ja     short pix_op_sub_w_3 ; 32bpp
7708 0000E3C7 7207                      <1>       jb     short pix_op_sub_w_2 ; 16bpp
7709                                    <1>
7710                                    <1>       ; 24 bit true colors
7711 0000E3C9 BD[71ED0000]              <1>       mov    ebp, pix_op_sub_24
7712 0000E3CE EB0C                      <1>       jmp    short pix_op_sub_w_4
7713                                    <1>
7714                                    <1>       ; 65536 colors (16bpp)
7715                                    <1> pix_op_sub_w_2:
7716 0000E3D0 BD[5EED0000]              <1>       mov    ebp, pix_op_sub_16
7717 0000E3D5 EB05                      <1>       jmp    short pix_op_sub_w_4
7718                                    <1>
7719                                    <1>       ; 32 bit true colors
7720                                    <1> pix_op_sub_w_3:
7721 0000E3D7 BD[8BED0000]              <1>       mov    ebp, pix_op_sub_32
7722                                    <1> pix_op_sub_w_4:
7723 0000E3DC E9AD000000                <1>       jmp    pix_op_sub_w_x
7724                                    <1>
7725                                    <1> pix_op_mix:
7726                                    <1>       ; 31/01/2021
7727                                    <1>       ; MIX COLOR
7728                                    <1>       ;
7729                                    <1>       ; INPUT:
7730                                    <1>       ;   CL = color (8 bit, 256 colors)
7731                                    <1>       ;  ECX = color (16 bit and true colors)
7732                                    <1>       ;  EDX = start position (row, column)
7733                                    <1>       ;        (HW = row, DX = column)
7734                                    <1>       ;  ESI = size (rows, colums)
7735                                    <1>       ;        (HW = rows, SI = columns)
7736                                    <1>       ;
7737                                    <1>       ;  [maskcolor] = mask color (to be excluded)
7738                                    <1>       ;
7739                                    <1>       ; OUTPUT:
7740                                    <1>       ;     [u.r0] will be > 0 if succesful
7741                                    <1>
7742 0000E3E1 F605[70120300]10          <1>       test   byte [v_ops], 10h ; display page or window ?
7743 0000E3E8 7555                      <1>       jnz    short pix_op_mix_w ; window
7744                                    <1>
7745 0000E3EA 8B3D[72120300]            <1>       mov    edi, [v_mem]
7746 0000E3F0 89FE                      <1>       mov    esi, edi
7747                                    <1>       ; ecx = color (CL, CX, ECX)
7748 0000E3F2 89C8                      <1>       mov    eax, ecx
7749 0000E3F4 8B0D[76120300]            <1>       mov    ecx, [v_siz] ; display page pixel count
7750                                    <1>
7751 0000E3FA F605[70120300]20          <1>       test   byte [v_ops], 20h ; masked color mix ?
7752 0000E401 7405                      <1>       jz     short pix_op_mix_0 ; no
7753 0000E403 E921110000                <1>       jmp    m_pix_op_mix ; mix colors except mask color
7754                                    <1> pix_op_mix_0:
7755 0000E408 803D[71120300]08          <1>       cmp    byte [v_bpp], 8 ; 8bpp
7756 0000E40F 7707                      <1>       ja     short pix_op_mix_1
7757                                    <1>
7758                                    <1>       ; 256 colors (8bpp)
7759 0000E411 E8F4090000                <1>       call   pix_op_mix_8
7760 0000E416 EB1E                      <1>       jmp    short pix_op_mix_4
7761                                    <1>
7762                                    <1> pix_op_mix_1:
7763 0000E418 803D[71120300]18          <1>       cmp    byte [v_bpp], 24 ; 24bpp
7764 0000E41F 7710                      <1>       ja     short pix_op_mix_3 ; 32bpp
7765 0000E421 7207                      <1>       jb     short pix_op_mix_2 ; 16bpp
7766                                    <1>
7767                                    <1>       ; 24 bit true colors
7768 0000E423 E8FD090000                <1>       call   pix_op_mix_24
7769 0000E428 EB0C                      <1>       jmp    short pix_op_mix_4
7770                                    <1>
7771                                    <1>       ; 65536 colors (16bpp)
7772                                    <1> pix_op_mix_2:
7773 0000E42A E8E7090000                <1>       call   pix_op_mix_16
```

```
7774 0000E42F EB05                <1>       jmp    short pix_op_mix_4
7775                              <1>
7776                              <1>       ; 32 bit true colors
7777                              <1> pix_op_mix_3:
7778 0000E431 E80B0A0000          <1>       call   pix_op_mix_32
7779                              <1> pix_op_mix_4:
7780 0000E436 29F7                <1>       sub    edi, esi
7781 0000E438 893D[64030300]      <1>       mov    [u.r0], edi
7782                              <1> pix_op_mix_5:
7783 0000E43E C3                  <1>       retn
7784                              <1>
7785                              <1> pix_op_mix_w:
7786                              <1>       ; 31/01/2021
7787 0000E43F 51                  <1>       push   ecx ; * ; color
7788 0000E440 89D1                <1>       mov    ecx, edx ; win start pos
7789 0000E442 89F2                <1>       mov    edx, esi ; size (rows, cols)
7790 0000E444 E8C4FCFFFF          <1>       call   sysvideo_15_12 ; window preparations
7791 0000E449 58                  <1>       pop    eax ; * ; color
7792 0000E44A 72F2                <1>       jc     short pix_op_mix_5
7793                              <1>
7794 0000E44C F605[70120300]20    <1>       test   byte [v_ops], 20h ; masked color mix ?
7795 0000E453 7405                <1>       jz     short pix_op_mix_w_0 ; no
7796 0000E455 E96C110000          <1>       jmp    m_pix_op_mix_w
7797                              <1>                      ; window mix colors except mask color
7798                              <1> pix_op_mix_w_0:
7799                              <1>       ; ecx = bytes per row (to be applied)
7800                              <1>       ; edx = screen width in bytes
7801                              <1>       ; ebx = row count
7802                              <1>       ; eax = color
7803                              <1>
7804 0000E45A 8B3D[7A120300]      <1>       mov    edi, [v_str]
7805 0000E460 803D[71120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
7806 0000E467 7707                <1>       ja     short pix_op_mix_w_1
7807                              <1>
7808                              <1>       ; 256 colors (8bpp)
7809 0000E469 BD[0AEE0000]        <1>       mov    ebp, pix_op_mix_8
7810 0000E46E EB1E                <1>       jmp    short pix_op_mix_w_x
7811                              <1>
7812                              <1> pix_op_mix_w_1:
7813 0000E470 803D[71120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
7814 0000E477 7710                <1>       ja     short pix_op_mix_w_3 ; 32bpp
7815 0000E479 7207                <1>       jb     short pix_op_mix_w_2 ; 16bpp
7816                              <1>
7817                              <1>       ; 24 bit true colors
7818 0000E47B BD[25EE0000]        <1>       mov    ebp, pix_op_mix_24
7819 0000E480 EB0C                <1>       jmp    short pix_op_mix_w_x
7820                              <1>
7821                              <1>       ; 65536 colors (16bpp)
7822                              <1> pix_op_mix_w_2:
7823 0000E482 BD[16EE0000]        <1>       mov    ebp, pix_op_mix_16
7824 0000E487 EB05                <1>       jmp    short pix_op_mix_w_x
7825                              <1>
7826                              <1>       ; 32 bit true colors
7827                              <1> pix_op_mix_w_3:
7828 0000E489 BD[41EE0000]        <1>       mov    ebp, pix_op_mix_32
7829                              <1>       ;jmp   short pix_op_mix_w_x
7830                              <1>
7831                              <1> pix_op_mix_w_x:
7832                              <1> pix_op_add_w_x:
7833                              <1> pix_op_sub_w_x:
7834                              <1> pix_op_rpl_w_x:
7835                              <1> pix_op_orc_w_x:
7836                              <1> pix_op_and_w_x:
7837                              <1> pix_op_xor_w_x:
7838                              <1>       ; 27/02/2021
7839                              <1>       ; 31/01/2021
7840                              <1>       ; ecx = bytes per row (to be applied)
7841                              <1>       ; edx = windows (screen) width in bytes
7842                              <1>       ; ebx = row count
7843                              <1>       ; eax = color
7844                              <1>       ; ebp = pixel operation subroutine address
7845 0000E48E 52                  <1>       push   edx
7846 0000E48F 51                  <1>       push   ecx
7847 0000E490 57                  <1>       push   edi
7848 0000E491 8B0D[86120300]      <1>       mov    ecx, [pixcount] ; 27/02/2021
7849 0000E497 FFD5                <1>       call   ebp ; call pixel-row operation
7850 0000E499 5F                  <1>       pop    edi
7851 0000E49A 59                  <1>       pop    ecx ; bytes per row
7852 0000E49B 010D[64030300]      <1>       add    [u.r0], ecx
7853 0000E4A1 5A                  <1>       pop    edx
7854 0000E4A2 01D7                <1>       add    edi, edx ; next row
7855 0000E4A4 4B                  <1>       dec    ebx
7856 0000E4A5 75E7                <1>       jnz    short pix_op_mix_w_x
7857 0000E4A7 C3                  <1>       retn
7858                              <1>
7859                              <1> pix_op_rpl:
7860                              <1>       ; 01/02/2021
7861                              <1>       ; REPLACE COLOR
7862                              <1>       ;
7863                              <1>       ; INPUT:
7864                              <1>       ;   CL = old/current color (8 bit, 256 colors)
7865                              <1>       ;  ECX = old/current color (16 bit and true colors)
7866                              <1>       ;   DL = new color (8 bit, 256 colors)
7867                              <1>       ;  EDX = new color (16 bit and true colors)
7868                              <1>       ;  ESI = start position (row, column)
7869                              <1>       ;        (HW = row, DX = column)
7870                              <1>       ;  EDI = size (rows, colums)
7871                              <1>       ;        (HW = rows, SI = columns)
7872                              <1>       ; OUTPUT:
7873                              <1>       ;      [u.r0] will be > 0 if succesful
7874                              <1>
7875 0000E4A8 F605[70120300]10    <1>       test   byte [v_ops], 10h ; display page or window ?
7876 0000E4AF 754D                <1>       jnz    short pix_op_rpl_w ; window
7877                              <1>
7878 0000E4B1 8B3D[72120300]      <1>       mov    edi, [v_mem]
```

```
7879 0000E4B7 89FE                <1>       mov    esi, edi
7880                              <1>       ; ecx = old color (CL, CX, ECX) -to be replaced with-
7881                              <1>       ; edx = new color (CL, CX, ECX) -new one-
7882 0000E4B9 89D0                <1>       mov    eax, edx ; new color
7883 0000E4BB 890D[82120300]      <1>       mov    [maskcolor], ecx ; old color
7884 0000E4C1 8B0D[76120300]      <1>       mov    ecx, [v_siz] ; display page pixel count
7885                              <1> pix_op_rpl_0:
7886 0000E4C7 803D[71120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
7887 0000E4CE 7707                <1>       ja     short pix_op_rpl_1
7888                              <1>
7889                              <1>       ; 256 colors (8bpp)
7890 0000E4D0 E8300A0000          <1>       call   pix_op_rpl_8
7891 0000E4D5 EB1E                <1>       jmp    short pix_op_rpl_4
7892                              <1>
7893                              <1> pix_op_rpl_1:
7894 0000E4D7 803D[71120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
7895 0000E4DE 7710                <1>       ja     short pix_op_rpl_3 ; 32bpp
7896 0000E4E0 7207                <1>       jb     short pix_op_rpl_2 ; 16bpp
7897                              <1>
7898                              <1>       ; 24 bit true colors
7899 0000E4E2 E8410A0000          <1>       call   pix_op_rpl_24
7900 0000E4E7 EB0C                <1>       jmp    short pix_op_rpl_4
7901                              <1>
7902                              <1>       ; 65536 colors (16bpp)
7903                              <1> pix_op_rpl_2:
7904 0000E4E9 E8270A0000          <1>       call   pix_op_rpl_16
7905 0000E4EE EB05                <1>       jmp    short pix_op_rpl_4
7906                              <1>
7907                              <1>       ; 32 bit true colors
7908                              <1> pix_op_rpl_3:
7909 0000E4F0 E8550A0000          <1>       call   pix_op_rpl_32
7910                              <1> pix_op_rpl_4:
7911 0000E4F5 29F7                <1>       sub    edi, esi
7912 0000E4F7 893D[64030300]      <1>       mov    [u.r0], edi
7913                              <1> pix_op_rpl_5:
7914 0000E4FD C3                  <1>       retn
7915                              <1>
7916                              <1> pix_op_rpl_w:
7917                              <1>       ; 01/02/2021
7918 0000E4FE 890D[82120300]      <1>       mov    [maskcolor], ecx ; old color
7919 0000E504 52                  <1>       push   edx ; * ; new color
7920 0000E505 89F1                <1>       mov    ecx, esi ; win start pos
7921 0000E507 89FA                <1>       mov    edx, edi ; size (rows, cols)
7922 0000E509 E8FFFBFFFF          <1>       call   sysvideo_15_12 ; window preparations
7923 0000E50E 58                  <1>       pop    eax ; * ; new color
7924 0000E50F 72EC                <1>       jc     short pix_op_rpl_5
7925                              <1>
7926                              <1>       ; replace window color
7927                              <1> pix_op_rpl_w_0:
7928                              <1>       ; ecx = bytes per row (to be applied)
7929                              <1>       ; edx = screen width in bytes
7930                              <1>       ; ebx = row count
7931                              <1>       ; eax = new color
7932                              <1>       ; [maskcolor] = old color
7933                              <1>
7934 0000E511 8B3D[7A120300]      <1>       mov    edi, [v_str]
7935                              <1>
7936 0000E517 803D[71120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
7937 0000E51E 7707                <1>       ja     short pix_op_rpl_w_1
7938                              <1>
7939                              <1>       ; 256 colors (8bpp)
7940 0000E520 BD[05EF0000]        <1>       mov    ebp, pix_op_rpl_8
7941 0000E525 EB1E                <1>       jmp    short pix_op_rpl_w_4
7942                              <1>
7943                              <1> pix_op_rpl_w_1:
7944 0000E527 803D[71120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
7945 0000E52E 7710                <1>       ja     short pix_op_rpl_w_3 ; 32bpp
7946 0000E530 7207                <1>       jb     short pix_op_rpl_w_2 ; 16bpp
7947                              <1>
7948                              <1>       ; 24 bit true colors
7949 0000E532 BD[28EF0000]        <1>       mov    ebp, pix_op_rpl_24
7950 0000E537 EB0C                <1>       jmp    short pix_op_rpl_w_4
7951                              <1>
7952                              <1>       ; 65536 colors (16bpp)
7953                              <1> pix_op_rpl_w_2:
7954 0000E539 BD[15EF0000]        <1>       mov    ebp, pix_op_rpl_16
7955 0000E53E EB05                <1>       jmp    short pix_op_rpl_w_4
7956                              <1>
7957                              <1>       ; 32 bit true colors
7958                              <1> pix_op_rpl_w_3:
7959 0000E540 BD[4AEF0000]        <1>       mov    ebp, pix_op_rpl_32
7960                              <1> pix_op_rpl_w_4:
7961 0000E545 E944FFFFFF          <1>       jmp    pix_op_rpl_w_x
7962                              <1>
7963                              <1> pix_op_orc:
7964                              <1>       ; 31/01/2021
7965                              <1>       ; OR COLOR
7966                              <1>       ;
7967                              <1>       ; INPUT:
7968                              <1>       ;   CL = color (8 bit, 256 colors)
7969                              <1>       ;  ECX = color (16 bit and true colors)
7970                              <1>       ;  EDX = start position (row, column)
7971                              <1>       ;        (HW = row, DX = column)
7972                              <1>       ;  ESI = size (rows, colums)
7973                              <1>       ;        (HW = rows, SI = columns)
7974                              <1>       ;
7975                              <1>       ;  [maskcolor] = mask color (to be excluded)
7976                              <1>       ;
7977                              <1>       ; OUTPUT:
7978                              <1>       ;      [u.r0] will be > 0 if succesful
7979                              <1>
7980 0000E54A F605[70120300]10    <1>       test   byte [v_ops], 10h ; display page or window ?
7981 0000E551 7555                <1>       jnz    short pix_op_or_w ; window
7982                              <1>
7983 0000E553 8B3D[72120300]      <1>       mov    edi, [v_mem]
```

```
7984 0000E559 89FE              <1>       mov    esi, edi
7985                            <1>       ; ecx = color (CL, CX, ECX)
7986 0000E55B 89C8              <1>       mov    eax, ecx
7987 0000E55D 8B0D[76120300]    <1>       mov    ecx, [v_siz] ; display page pixel count
7988                            <1>
7989 0000E563 F605[70120300]20  <1>       test   byte [v_ops], 20h ; masked color 'or' ?
7990 0000E56A 7405              <1>       jz     short pix_op_or_0 ; no
7991 0000E56C E948110000        <1>       jmp    m_pix_op_or ; 'or' color except mask color
7992                            <1> pix_op_or_0:
7993 0000E571 803D[71120300]08  <1>       cmp    byte [v_bpp], 8 ; 8bpp
7994 0000E578 7707              <1>       ja     short pix_op_or_1
7995                            <1>
7996                            <1>       ; 256 colors (8bpp)
7997 0000E57A E81C080000        <1>       call   pix_op_or_8
7998 0000E57F EB1E              <1>       jmp    short pix_op_or_4
7999                            <1>
8000                            <1> pix_op_or_1:
8001 0000E581 803D[71120300]18  <1>       cmp    byte [v_bpp], 24 ; 24bpp
8002 0000E588 7710              <1>       ja     short pix_op_or_3 ; 32bpp
8003 0000E58A 7207              <1>       jb     short pix_op_or_2 ; 16bpp
8004                            <1>
8005                            <1>       ; 24 bit true colors
8006 0000E58C E818080000        <1>       call   pix_op_or_24
8007 0000E591 EB0C              <1>       jmp    short pix_op_or_4
8008                            <1>
8009                            <1>       ; 65536 colors (16bpp)
8010                            <1> pix_op_or_2:
8011 0000E593 E809080000        <1>       call   pix_op_or_16
8012 0000E598 EB05              <1>       jmp    short pix_op_or_4
8013                            <1>
8014                            <1>       ; 32 bit true colors
8015                            <1> pix_op_or_3:
8016 0000E59A E819080000        <1>       call   pix_op_or_32
8017                            <1> pix_op_or_4:
8018 0000E59F 29F7              <1>       sub    edi, esi
8019 0000E5A1 893D[64030300]    <1>       mov    [u.r0], edi
8020                            <1> pix_op_or_5:
8021 0000E5A7 C3                <1>       retn
8022                            <1>
8023                            <1> pix_op_or_w:
8024                            <1>       ; 31/01/2021
8025 0000E5A8 51                <1>       push   ecx ; * ; color
8026 0000E5A9 89D1              <1>       mov    ecx, edx ; win start pos
8027 0000E5AB 89F2              <1>       mov    edx, esi ; size (rows, cols)
8028 0000E5AD E85BFBFFFF        <1>       call   sysvideo_15_12 ; window preparations
8029 0000E5B2 58                <1>       pop    eax ; * ; color
8030 0000E5B3 72F2              <1>       jc     short pix_op_or_5
8031                            <1>
8032 0000E5B5 F605[70120300]20  <1>       test   byte [v_ops], 20h ; masked color 'or' ?
8033 0000E5BC 7405              <1>       jz     short pix_op_or_w_0 ; no
8034 0000E5BE E983110000        <1>       jmp    m_pix_op_or_w
8035                            <1>                    ; window 'or' color except mask color
8036                            <1> pix_op_or_w_0:
8037                            <1>       ; ecx = bytes per row (to be applied)
8038                            <1>       ; edx = screen width in bytes
8039                            <1>       ; ebx = row count
8040                            <1>       ; eax = color
8041                            <1>
8042 0000E5C3 8B3D[7A120300]    <1>       mov    edi, [v_str]
8043 0000E5C9 803D[71120300]08  <1>       cmp    byte [v_bpp], 8 ; 8bpp
8044 0000E5D0 7707              <1>       ja     short pix_op_or_w_1
8045                            <1>
8046                            <1>       ; 256 colors (8bpp)
8047 0000E5D2 BD[9BED0000]      <1>       mov    ebp, pix_op_or_8
8048 0000E5D7 EB1E              <1>       jmp    short pix_op_or_w_4
8049                            <1>
8050                            <1> pix_op_or_w_1:
8051 0000E5D9 803D[71120300]18  <1>       cmp    byte [v_bpp], 24 ; 24bpp
8052 0000E5E0 7710              <1>       ja     short pix_op_or_w_3 ; 32bpp
8053 0000E5E2 7207              <1>       jb     short pix_op_or_w_2 ; 16bpp
8054                            <1>
8055                            <1>       ; 24 bit true colors
8056 0000E5E4 BD[A9ED0000]      <1>       mov    ebp, pix_op_or_24
8057 0000E5E9 EB0C              <1>       jmp    short pix_op_or_w_4
8058                            <1>
8059                            <1>       ; 65536 colors (16bpp)
8060                            <1> pix_op_or_w_2:
8061 0000E5EB BD[A1ED0000]      <1>       mov    ebp, pix_op_or_16
8062 0000E5F0 EB05              <1>       jmp    short pix_op_or_w_4
8063                            <1>
8064                            <1>       ; 32 bit true colors
8065                            <1> pix_op_or_w_3:
8066 0000E5F2 BD[B8ED0000]      <1>       mov    ebp, pix_op_or_32
8067                            <1> pix_op_or_w_4:
8068 0000E5F7 E992FEFFFF        <1>       jmp    pix_op_orc_w_x
8069                            <1>
8070                            <1> pix_op_and:
8071                            <1>       ; 31/01/2021
8072                            <1>       ; AND COLOR
8073                            <1>       ;
8074                            <1>       ; INPUT:
8075                            <1>       ;   CL = color (8 bit, 256 colors)
8076                            <1>       ;  ECX = color (16 bit and true colors)
8077                            <1>       ;  EDX = start position (row, column)
8078                            <1>       ;         (HW = row, DX = column)
8079                            <1>       ;  ESI = size (rows, colums)
8080                            <1>       ;         (HW = rows, SI = columns)
8081                            <1>       ;
8082                            <1>       ;  [maskcolor] = mask color (to be excluded)
8083                            <1>       ;
8084                            <1>       ; OUTPUT:
8085                            <1>       ;      [u.r0] will be > 0 if succesful
8086                            <1>
8087 0000E5FC F605[70120300]10  <1>       test   byte [v_ops], 10h ; display page or window ?
8088 0000E603 7555              <1>       jnz    short pix_op_and_w ; window
```

```
8089                                <1>
8090 0000E605 8B3D[72120300]        <1>       mov    edi, [v_mem]
8091 0000E60B 89FE                  <1>       mov    esi, edi
8092                                <1>       ; ecx = color (CL, CX, ECX)
8093 0000E60D 89C8                  <1>       mov    eax, ecx
8094 0000E60F 8B0D[76120300]        <1>       mov    ecx, [v_siz] ; display page pixel count
8095                                <1>
8096 0000E615 F605[70120300]20      <1>       test   byte [v_ops], 20h ; masked color 'and' ?
8097 0000E61C 7405                  <1>       jz     short pix_op_and_0 ; no
8098 0000E61E E9D60F0000            <1>       jmp    m_pix_op_and ; 'and' color except mask color
8099                                <1> pix_op_and_0:
8100 0000E623 803D[71120300]08      <1>       cmp    byte [v_bpp], 8 ; 8bpp
8101 0000E62A 7707                  <1>       ja     short pix_op_and_1
8102                                <1>
8103                                <1>       ; 256 colors (8bpp)
8104 0000E62C E88F070000            <1>       call   pix_op_and_8
8105 0000E631 EB1E                  <1>       jmp    short pix_op_and_4
8106                                <1>
8107                                <1> pix_op_and_1:
8108 0000E633 803D[71120300]18      <1>       cmp    byte [v_bpp], 24 ; 24bpp
8109 0000E63A 7710                  <1>       ja     short pix_op_and_3 ; 32bpp
8110 0000E63C 7207                  <1>       jb     short pix_op_and_2 ; 16bpp
8111                                <1>
8112                                <1>       ; 24 bit true colors
8113 0000E63E E88B070000            <1>       call   pix_op_and_24
8114 0000E643 EB0C                  <1>       jmp    short pix_op_and_4
8115                                <1>
8116                                <1>       ; 65536 colors (16bpp)
8117                                <1> pix_op_and_2:
8118 0000E645 E87C070000            <1>       call   pix_op_and_16
8119 0000E64A EB05                  <1>       jmp    short pix_op_and_4
8120                                <1>
8121                                <1>       ; 32 bit true colors
8122                                <1> pix_op_and_3:
8123 0000E64C E88C070000            <1>       call   pix_op_and_32
8124                                <1> pix_op_and_4:
8125 0000E651 29F7                  <1>       sub    edi, esi
8126 0000E653 893D[64030300]        <1>       mov    [u.r0], edi
8127                                <1> pix_op_and_5:
8128 0000E659 C3                    <1>       retn
8129                                <1>
8130                                <1> pix_op_and_w:
8131                                <1>       ; 31/01/2021
8132 0000E65A 51                    <1>       push   ecx ; * ; color
8133 0000E65B 89D1                  <1>       mov    ecx, edx ; win start pos
8134 0000E65D 89F2                  <1>       mov    edx, esi ; size (rows, cols)
8135 0000E65F E8A9FAFFFF            <1>       call   sysvideo_15_12 ; window preparations
8136 0000E664 58                    <1>       pop    eax ; * ; color
8137 0000E665 72F2                  <1>       jc     short pix_op_and_5
8138                                <1>
8139 0000E667 F605[70120300]20      <1>       test   byte [v_ops], 20h ; masked color 'and' ?
8140 0000E66E 7405                  <1>       jz     short pix_op_and_w_0 ; no
8141 0000E670 E911100000            <1>       jmp    m_pix_op_and_w
8142                                <1>                       ; window 'and' color except mask color
8143                                <1> pix_op_and_w_0:
8144                                <1>       ; ecx = bytes per row (to be applied)
8145                                <1>       ; edx = screen width in bytes
8146                                <1>       ; ebx = row count
8147                                <1>       ; eax = color
8148                                <1>
8149 0000E675 8B3D[7A120300]        <1>       mov    edi, [v_str]
8150 0000E67B 803D[71120300]08      <1>       cmp    byte [v_bpp], 8 ; 8bpp
8151 0000E682 7707                  <1>       ja     short pix_op_and_w_1
8152                                <1>
8153                                <1>       ; 256 colors (8bpp)
8154 0000E684 BD[C0ED0000]          <1>       mov    ebp, pix_op_and_8
8155 0000E689 EB1E                  <1>       jmp    short pix_op_and_w_4
8156                                <1>
8157                                <1> pix_op_and_w_1:
8158 0000E68B 803D[71120300]18      <1>       cmp    byte [v_bpp], 24 ; 24bpp
8159 0000E692 7710                  <1>       ja     short pix_op_and_w_3 ; 32bpp
8160 0000E694 7207                  <1>       jb     short pix_op_and_w_2 ; 16bpp
8161                                <1>
8162                                <1>       ; 24 bit true colors
8163 0000E696 BD[CEED0000]          <1>       mov    ebp, pix_op_and_24
8164 0000E69B EB0C                  <1>       jmp    short pix_op_and_w_4
8165                                <1>
8166                                <1>       ; 65536 colors (16bpp)
8167                                <1> pix_op_and_w_2:
8168 0000E69D BD[C6ED0000]          <1>       mov    ebp, pix_op_and_16
8169 0000E6A2 EB05                  <1>       jmp    short pix_op_and_w_4
8170                                <1>
8171                                <1>       ; 32 bit true colors
8172                                <1> pix_op_and_w_3:
8173 0000E6A4 BD[DDED0000]          <1>       mov    ebp, pix_op_and_32
8174                                <1> pix_op_and_w_4:
8175 0000E6A9 E9E0FDFFFF            <1>       jmp    pix_op_and_w_x
8176                                <1>
8177                                <1> pix_op_xor:
8178                                <1>       ; 31/01/2021
8179                                <1>       ; XOR COLOR
8180                                <1>       ;
8181                                <1>       ; INPUT:
8182                                <1>       ;   CL = color (8 bit, 256 colors)
8183                                <1>       ;  ECX = color (16 bit and true colors)
8184                                <1>       ;  EDX = start position (row, column)
8185                                <1>       ;        (HW = row, DX = column)
8186                                <1>       ;  ESI = size (rows, colums)
8187                                <1>       ;        (HW = rows, SI = columns)
8188                                <1>       ;
8189                                <1>       ;  [maskcolor] = mask color (to be excluded)
8190                                <1>       ;
8191                                <1>       ; OUTPUT:
8192                                <1>       ;      [u.r0] will be > 0 if succesful
8193                                <1>
```

```
8194 0000E6AE F605[70120300]10    <1>       test   byte [v_ops], 10h ; display page or window ?
8195 0000E6B5 7555                <1>       jnz    short pix_op_xor_w ; window
8196                              <1>
8197 0000E6B7 8B3D[72120300]      <1>       mov    edi, [v_mem]
8198 0000E6BD 89FE                <1>       mov    esi, edi
8199                              <1>       ; ecx = color (CL, CX, ECX)
8200 0000E6BF 89C8                <1>       mov    eax, ecx
8201 0000E6C1 8B0D[76120300]      <1>       mov    ecx, [v_siz] ; display page pixel count
8202                              <1>
8203 0000E6C7 F605[70120300]20    <1>       test   byte [v_ops], 20h ; masked color 'xor' ?
8204 0000E6CE 7405                <1>       jz     short pix_op_xor_0 ; no
8205 0000E6D0 E9A4100000          <1>       jmp    m_pix_op_xor ; 'xor' color except mask color
8206                              <1> pix_op_xor_0:
8207 0000E6D5 803D[71120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
8208 0000E6DC 7707                <1>       ja     short pix_op_xor_1
8209                              <1>
8210                              <1>       ; 256 colors (8bpp)
8211 0000E6DE E802070000          <1>       call   pix_op_xor_8
8212 0000E6E3 EB1E                <1>       jmp    short pix_op_xor_4
8213                              <1>
8214                              <1> pix_op_xor_1:
8215 0000E6E5 803D[71120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
8216 0000E6EC 7710                <1>       ja     short pix_op_xor_3 ; 32bpp
8217 0000E6EE 7207                <1>       jb     short pix_op_xor_2 ; 16bpp
8218                              <1>
8219                              <1>       ; 24 bit true colors
8220 0000E6F0 E8FE060000          <1>       call   pix_op_xor_24
8221 0000E6F5 EB0C                <1>       jmp    short pix_op_xor_4
8222                              <1>
8223                              <1>       ; 65536 colors (16bpp)
8224                              <1> pix_op_xor_2:
8225 0000E6F7 E8EF060000          <1>       call   pix_op_xor_16
8226 0000E6FC EB05                <1>       jmp    short pix_op_xor_4
8227                              <1>
8228                              <1>       ; 32 bit true colors
8229                              <1> pix_op_xor_3:
8230 0000E6FE E8FF060000          <1>       call   pix_op_xor_32
8231                              <1> pix_op_xor_4:
8232 0000E703 29F7                <1>       sub    edi, esi
8233 0000E705 893D[64030300]      <1>       mov    [u.r0], edi
8234                              <1> pix_op_xor_5:
8235 0000E70B C3                  <1>       retn
8236                              <1>
8237                              <1> pix_op_xor_w:
8238                              <1>       ; 31/01/2021
8239 0000E70C 51                  <1>       push   ecx ; * ; color
8240 0000E70D 89D1                <1>       mov    ecx, edx ; win start pos
8241 0000E70F 89F2                <1>       mov    edx, esi ; size (rows, cols)
8242 0000E711 E8F7F9FFFF          <1>       call   sysvideo_15_12 ; window preparations
8243 0000E716 58                  <1>       pop    eax ; * ; color
8244 0000E717 72F2                <1>       jc     short pix_op_xor_5
8245                              <1>
8246 0000E719 F605[70120300]20    <1>       test   byte [v_ops], 20h ; masked color 'xor' ?
8247 0000E720 7405                <1>       jz     short pix_op_xor_w_0 ; no
8248 0000E722 E9DF100000          <1>       jmp    m_pix_op_xor_w
8249                              <1>                      ; window 'xor' color except mask color
8250                              <1> pix_op_xor_w_0:
8251                              <1>       ; ecx = bytes per row (to be applied)
8252                              <1>       ; edx = screen width in bytes
8253                              <1>       ; ebx = row count
8254                              <1>       ; eax = color
8255                              <1>
8256 0000E727 8B3D[7A120300]      <1>       mov    edi, [v_str]
8257 0000E72D 803D[71120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
8258 0000E734 7707                <1>       ja     short pix_op_xor_w_1
8259                              <1>
8260                              <1>       ; 256 colors (8bpp)
8261 0000E736 BD[E5ED0000]        <1>       mov    ebp, pix_op_xor_8
8262 0000E73B EB1E                <1>       jmp    short pix_op_xor_w_4
8263                              <1>
8264                              <1> pix_op_xor_w_1:
8265 0000E73D 803D[71120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
8266 0000E744 7710                <1>       ja     short pix_op_xor_w_3 ; 32bpp
8267 0000E746 7207                <1>       jb     short pix_op_xor_w_2 ; 16bpp
8268                              <1>
8269                              <1>       ; 24 bit true colors
8270 0000E748 BD[F3ED0000]        <1>       mov    ebp, pix_op_xor_24
8271 0000E74D EB0C                <1>       jmp    short pix_op_xor_w_4
8272                              <1>
8273                              <1>       ; 65536 colors (16bpp)
8274                              <1> pix_op_xor_w_2:
8275 0000E74F BD[EBED0000]        <1>       mov    ebp, pix_op_xor_16
8276 0000E754 EB05                <1>       jmp    short pix_op_xor_w_4
8277                              <1>
8278                              <1>       ; 32 bit true colors
8279                              <1> pix_op_xor_w_3:
8280 0000E756 BD[02EE0000]        <1>       mov    ebp, pix_op_xor_32
8281                              <1> pix_op_xor_w_4:
8282 0000E75B E92EFDFFFF          <1>       jmp    pix_op_xor_w_x
8283                              <1>
8284                              <1> pix_op_new:
8285                              <1>       ; 31/01/2021
8286                              <1>       ; 30/01/2021
8287                              <1>       ; CHANGE COLOR
8288                              <1>       ;
8289                              <1>       ; INPUT:
8290                              <1>       ;   CL = color (8 bit, 256 colors)
8291                              <1>       ;  ECX = color (16 bit and true colors)
8292                              <1>       ;  EDX = start position (row, column)
8293                              <1>       ;        (HW = row, DX = column)
8294                              <1>       ;  ESI = size (rows, colums)
8295                              <1>       ;        (HW = rows, SI = columns)
8296                              <1>       ;
8297                              <1>       ;  [maskcolor] = mask color (to be excluded)
8298                              <1>       ;
```

```
                                <1>       ; OUTPUT:
                                <1>       ;      [u.r0] will be > 0 if succesful
                                <1>
0000E760 F605[70120300]10       <1>       test   byte [v_ops], 10h ; display page or window ?
0000E767 7554                   <1>       jnz    short pix_op_new_w ; window
                                <1>
0000E769 8B3D[72120300]         <1>       mov    edi, [v_mem]
0000E76F 89FE                   <1>       mov    esi, edi
                                <1>       ; ecx = color (CL, CX, ECX)
0000E771 89C8                   <1>       mov    eax, ecx
0000E773 8B0D[76120300]         <1>       mov    ecx, [v_siz] ; display page pixel count
                                <1>
0000E779 F605[70120300]20       <1>       test   byte [v_ops], 20h ; masked color change ?
0000E780 7405                   <1>       jz     short pix_op_new_0 ; no
0000E782 E90D0B0000             <1>       jmp    m_pix_op_new ; change color except mask color
                                <1> pix_op_new_0:
0000E787 803D[71120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000E78E 7706                   <1>       ja     short pix_op_new_2
                                <1>
                                <1>       ; 256 colors (8bpp)
                                <1> pix_op_new_1:
0000E790 88C4                   <1>       mov    ah, al
0000E792 D1E9                   <1>       shr    ecx, 1
0000E794 EB12                   <1>       jmp    short pix_op_new_3
                                <1>
                                <1> pix_op_new_2:
0000E796 803D[71120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000E79D 7713                   <1>       ja     short pix_op_new_4 ; 32bpp
0000E79F 7207                   <1>       jb     short pix_op_new_3 ; 16bpp
                                <1>
                                <1>       ; 31/01/2021
                                <1>
                                <1>       ; 24 bit true colors
0000E7A1 E84A050000             <1>       call   pix_op_new_24
                                <1>
0000E7A6 EB0C                   <1>       jmp    short pix_op_new_5
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> pix_op_new_3:
0000E7A8 89C2                   <1>       mov    edx, eax
0000E7AA C1E010                 <1>       shl    eax, 16
0000E7AD 6689D0                 <1>       mov    ax, dx
0000E7B0 D1E9                   <1>       shr    ecx, 1 ; dword counts
                                <1>       ; 32 bit true colors
                                <1> pix_op_new_4:
0000E7B2 F3AB                   <1>       rep    stosd
                                <1> pix_op_new_5:
0000E7B4 29F7                   <1>       sub    edi, esi
0000E7B6 893D[64030300]         <1>       mov    [u.r0], edi
                                <1> pix_op_new_6:
0000E7BC C3                     <1>       retn
                                <1>
                                <1> pix_op_new_w:
                                <1>       ; 31/01/2021
                                <1>       ; 30/01/2021
0000E7BD 51                     <1>       push   ecx ; * ; color
0000E7BE 89D1                   <1>       mov    ecx, edx ; win start pos
0000E7C0 89F2                   <1>       mov    edx, esi ; size (rows, cols)
0000E7C2 E846F9FFFF             <1>       call   sysvideo_15_12 ; window preparations
0000E7C7 58                     <1>       pop    eax ; * ; color
0000E7C8 72F2                   <1>       jc     short pix_op_new_6
                                <1>
0000E7CA F605[70120300]20       <1>       test   byte [v_ops], 20h ; masked color change ?
0000E7D1 7405                   <1>       jz     short pix_op_new_w_0 ; no
0000E7D3 E94A0B0000             <1>       jmp    m_pix_op_new_w
                                <1>                      ; window chg color except mask color
                                <1> pix_op_new_w_0:
                                <1>       ; ecx = bytes per row (to be applied)
                                <1>       ; edx = screen width in bytes
                                <1>       ; ebx = row count
                                <1>       ; eax = color
                                <1>
0000E7D8 8B3D[7A120300]         <1>       mov    edi, [v_str]
                                <1>
0000E7DE 803D[71120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000E7E5 7707                   <1>       ja     short pix_op_new_w_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0000E7E7 BD[E9EC0000]           <1>       mov    ebp, pix_op_new_8
0000E7EC EB1E                   <1>       jmp    short pix_op_new_w_x
                                <1>
                                <1> pix_op_new_w_1:
0000E7EE 803D[71120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000E7F5 7710                   <1>       ja     short pix_op_new_w_3 ; 32bpp
0000E7F7 7207                   <1>       jb     short pix_op_new_w_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0000E7F9 BD[F0EC0000]           <1>       mov    ebp, pix_op_new_24
0000E7FE EB0C                   <1>       jmp    short pix_op_new_w_x
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> pix_op_new_w_2:
0000E800 BD[ECEC0000]           <1>       mov    ebp, pix_op_new_16
0000E805 EB05                   <1>       jmp    short pix_op_new_w_x
                                <1>
                                <1>       ; 32 bit true colors
                                <1> pix_op_new_w_3:
0000E807 BD[FCEC0000]           <1>       mov    ebp, pix_op_new_32
                                <1>       ;jmp   short pix_op_new_w_x
                                <1>
                                <1> pix_op_new_w_x:
                                <1> pix_op_not_w_x:
                                <1> pix_op_neg_w_x:
                                <1> pix_op_inc_w_x:
                                <1> pix_op_dec_w_x:
```

```
                                <1>       ; 27/02/2021
                                <1>       ; 01/02/2021
                                <1>       ; 31/01/2021
                                <1>       ; ecx = bytes per row (to be applied)
                                <1>       ; edx = windows (screen) width in bytes
                                <1>       ; ebx = row count
                                <1>       ; eax = color
                                <1>       ; ebp = pixel operation subroutine address
                                <1>       ;push  edx ; 01/02/2021
0000E80C 51                     <1>       push   ecx
0000E80D 57                     <1>       push   edi
0000E80E 8B0D[86120300]         <1>       mov    ecx, [pixcount] ; 27/02/2021
0000E814 FFD5                   <1>       call   ebp ; call pixel-row operation
0000E816 5F                     <1>       pop    edi
0000E817 59                     <1>       pop    ecx ; bytes per row
0000E818 010D[64030300]         <1>       add    [u.r0], ecx
                                <1>       ;pop   edx ; 01/02/2021
0000E81E 01D7                   <1>       add    edi, edx ; next row
0000E820 4B                     <1>       dec    ebx
0000E821 75E9                   <1>       jnz    short pix_op_new_w_x
0000E823 C3                     <1>       retn
                                <1>
                                <1> pix_op_not:
                                <1>       ; 31/01/2021
                                <1>       ; NOT COLOR
                                <1>       ;
                                <1>       ; INPUT:
                                <1>       ;  ECX = start position (row, column)
                                <1>       ;         (HW = row, CX = column)
                                <1>       ;  EDX = size (rows, colums)
                                <1>       ;         (HW = rows, DX = columns)
                                <1>       ;       (0 -> invalid
                                <1>       ;         (1 -> horizontal or vertical line)
                                <1>       ;  [maskcolor] = mask color (to be excluded)
                                <1>       ;
                                <1>       ; OUTPUT:
                                <1>       ;      [u.r0] will be > 0 if succesful
                                <1>
0000E824 F605[70120300]10       <1>       test   byte [v_ops], 10h ; display page or window ?
0000E82B 7553                   <1>       jnz    short pix_op_not_w ; window
                                <1>
0000E82D 8B3D[72120300]         <1>       mov    edi, [v_mem]
0000E833 89FE                   <1>       mov    esi, edi
0000E835 8B0D[76120300]         <1>       mov    ecx, [v_siz] ; display page pixel count
                                <1>
0000E83B F605[70120300]20       <1>       test   byte [v_ops], 20h ; masked color 'not' ?
0000E842 7405                   <1>       jz     short pix_op_not_0 ; no
0000E844 E9F00F0000             <1>       jmp    m_pix_op_not ; 'not' color except mask color
                                <1> pix_op_not_0:
0000E849 803D[71120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000E850 7707                   <1>       ja     short pix_op_not_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0000E852 E8F6050000             <1>       call   pix_op_not_8
0000E857 EB1E                   <1>       jmp    short pix_op_not_4
                                <1>
                                <1> pix_op_not_1:
0000E859 803D[71120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000E860 7710                   <1>       ja     short pix_op_not_3 ; 32bpp
0000E862 7207                   <1>       jb     short pix_op_not_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0000E864 E8F2050000             <1>       call   pix_op_not_24
0000E869 EB0C                   <1>       jmp    short pix_op_not_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> pix_op_not_2:
0000E86B E8E3050000             <1>       call   pix_op_not_16
0000E870 EB05                   <1>       jmp    short pix_op_not_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> pix_op_not_3:
0000E872 E8EF050000             <1>       call   pix_op_not_32
                                <1> pix_op_not_4:
0000E877 29F7                   <1>       sub    edi, esi
0000E879 893D[64030300]         <1>       mov    [u.r0], edi
                                <1> pix_op_not_5:
0000E87F C3                     <1>       retn
                                <1>
                                <1> pix_op_not_w:
                                <1>       ; 31/01/2021
                                <1>       ; ecx = win start pos (row, column)
                                <1>       ; edx = size (rows, columns)
0000E880 E888F8FFFF             <1>       call   sysvideo_15_12 ; window preparations
0000E885 72F8                   <1>       jc     short pix_op_not_5
                                <1>
0000E887 F605[70120300]20       <1>       test   byte [v_ops], 20h ; masked color 'not' ?
0000E88E 7405                   <1>       jz     short pix_op_not_w_0 ; no
0000E890 E929100000             <1>       jmp    m_pix_op_not_w
                                <1>                      ; window 'not' color except mask color
                                <1> pix_op_not_w_0:
                                <1>       ; ecx = bytes per row (to be applied)
                                <1>       ; edx = screen width in bytes
                                <1>       ; ebx = row count
                                <1>
0000E895 8B3D[7A120300]         <1>       mov    edi, [v_str]
                                <1>
0000E89B 803D[71120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000E8A2 7707                   <1>       ja     short pix_op_not_w_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0000E8A4 BD[4DEE0000]           <1>       mov    ebp, pix_op_not_8
0000E8A9 EB1E                   <1>       jmp    short pix_op_not_w_4
                                <1>
                                <1> pix_op_not_w_1:
```

```
8509 0000E8AB 803D[71120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
8510 0000E8B2 7710                <1>       ja     short pix_op_not_w_3 ; 32bpp
8511 0000E8B4 7207                <1>       jb     short pix_op_not_w_2 ; 16bpp
8512                              <1>
8513                              <1>       ; 24 bit true colors
8514 0000E8B6 BD[5BEE0000]        <1>       mov    ebp, pix_op_not_24
8515 0000E8BB EB0C                <1>       jmp    short pix_op_not_w_4
8516                              <1>
8517                              <1>       ; 65536 colors (16bpp)
8518                              <1> pix_op_not_w_2:
8519 0000E8BD BD[53EE0000]        <1>       mov    ebp, pix_op_not_16
8520 0000E8C2 EB05                <1>       jmp    short pix_op_not_w_4
8521                              <1>
8522                              <1>       ; 32 bit true colors
8523                              <1> pix_op_not_w_3:
8524 0000E8C4 BD[66EE0000]        <1>       mov    ebp, pix_op_not_32
8525                              <1> pix_op_not_w_4:
8526 0000E8C9 E93EFFFFFF          <1>       jmp    pix_op_not_w_x
8527                              <1>
8528                              <1> pix_op_neg:
8529                              <1>       ; 31/01/2021
8530                              <1>       ; NEGATE COLOR
8531                              <1>       ;
8532                              <1>       ; INPUT:
8533                              <1>       ;  ECX = start position (row, column)
8534                              <1>       ;         (HW = row, CX = column)
8535                              <1>       ;  EDX = size (rows, colums)
8536                              <1>       ;         (HW = rows, DX = columns)
8537                              <1>       ;       (0 -> invalid
8538                              <1>       ;         (1 -> horizontal or vertical line)
8539                              <1>       ;  [maskcolor] = mask color (to be excluded)
8540                              <1>       ;
8541                              <1>       ; OUTPUT:
8542                              <1>       ;      [u.r0] will be > 0 if succesful
8543                              <1>
8544 0000E8CE F605[70120300]10    <1>       test   byte [v_ops], 10h ; display page or window ?
8545 0000E8D5 7553                <1>       jnz    short pix_op_neg_w ; window
8546                              <1>
8547 0000E8D7 8B3D[72120300]      <1>       mov    edi, [v_mem]
8548 0000E8DD 89FE                <1>       mov    esi, edi
8549 0000E8DF 8B0D[76120300]      <1>       mov    ecx, [v_siz] ; display page pixel count
8550                              <1>
8551 0000E8E5 F605[70120300]20    <1>       test   byte [v_ops], 20h ; masked negate color ?
8552 0000E8EC 7405                <1>       jz     short pix_op_neg_0 ; no
8553 0000E8EE E9FE0F0000          <1>       jmp    m_pix_op_neg ; 'neg' color except mask color
8554                              <1> pix_op_neg_0:
8555 0000E8F3 803D[71120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
8556 0000E8FA 7707                <1>       ja     short pix_op_neg_1
8557                              <1>
8558                              <1>       ; 256 colors (8bpp)
8559 0000E8FC E86D050000          <1>       call   pix_op_neg_8
8560 0000E901 EB1E                <1>       jmp    short pix_op_neg_4
8561                              <1>
8562                              <1> pix_op_neg_1:
8563 0000E903 803D[71120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
8564 0000E90A 7710                <1>       ja     short pix_op_neg_3 ; 32bpp
8565 0000E90C 7207                <1>       jb     short pix_op_neg_2 ; 16bpp
8566                              <1>
8567                              <1>       ; 24 bit true colors
8568 0000E90E E869050000          <1>       call   pix_op_neg_24
8569 0000E913 EB0C                <1>       jmp    short pix_op_neg_4
8570                              <1>
8571                              <1>       ; 65536 colors (16bpp)
8572                              <1> pix_op_neg_2:
8573 0000E915 E85A050000          <1>       call   pix_op_neg_16
8574 0000E91A EB05                <1>       jmp    short pix_op_neg_4
8575                              <1>
8576                              <1>       ; 32 bit true colors
8577                              <1> pix_op_neg_3:
8578 0000E91C E86D050000          <1>       call   pix_op_neg_32
8579                              <1> pix_op_neg_4:
8580 0000E921 29F7                <1>       sub    edi, esi
8581 0000E923 893D[64030300]      <1>       mov    [u.r0], edi
8582                              <1> pix_op_neg_5:
8583 0000E929 C3                  <1>       retn
8584                              <1>
8585                              <1> pix_op_neg_w:
8586                              <1>       ; 31/01/2021
8587                              <1>       ; ecx = win start pos (row, column)
8588                              <1>       ; edx = size (rows, columns)
8589 0000E92A E8DEF7FFFF          <1>       call   sysvideo_15_12 ; window preparations
8590 0000E92F 72F8                <1>       jc     short pix_op_neg_5
8591                              <1>
8592 0000E931 F605[70120300]20    <1>       test   byte [v_ops], 20h ; masked negate color ?
8593 0000E938 7405                <1>       jz     short pix_op_neg_w_0 ; no
8594 0000E93A E937100000          <1>       jmp    m_pix_op_neg_w
8595                              <1>                     ; window 'neg' color except mask color
8596                              <1> pix_op_neg_w_0:
8597                              <1>       ; ecx = bytes per row (to be applied)
8598                              <1>       ; edx = screen width in bytes
8599                              <1>       ; ebx = row count
8600                              <1>
8601 0000E93F 8B3D[7A120300]      <1>       mov    edi, [v_str]
8602                              <1>
8603 0000E945 803D[71120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
8604 0000E94C 7707                <1>       ja     short pix_op_neg_w_1
8605                              <1>
8606                              <1>       ; 256 colors (8bpp)
8607 0000E94E BD[6EEE0000]        <1>       mov    ebp, pix_op_neg_8
8608 0000E953 EB1E                <1>       jmp    short pix_op_neg_w_4
8609                              <1>
8610                              <1> pix_op_neg_w_1:
8611 0000E955 803D[71120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
8612 0000E95C 7710                <1>       ja     short pix_op_neg_w_3 ; 32bpp
8613 0000E95E 7207                <1>       jb     short pix_op_neg_w_2 ; 16bpp
```

```
                                <1>
                                <1>       ; 24 bit true colors
0000E960 BD[7CEE0000]           <1>       mov    ebp, pix_op_neg_24
0000E965 EB0C                   <1>       jmp    short pix_op_neg_w_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> pix_op_neg_w_2:
0000E967 BD[74EE0000]           <1>       mov    ebp, pix_op_neg_16
0000E96C EB05                   <1>       jmp    short pix_op_neg_w_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> pix_op_neg_w_3:
0000E96E BD[8EEE0000]           <1>       mov    ebp, pix_op_neg_32
                                <1> pix_op_neg_w_4:
0000E973 E994FEFFFF             <1>       jmp    pix_op_neg_w_x
                                <1>
                                <1> pix_op_inc:
                                <1>       ; 31/01/2021
                                <1>       ; INCREASE COLOR
                                <1>       ;
                                <1>       ; INPUT:
                                <1>       ;  ECX = start position (row, column)
                                <1>       ;         (HW = row, CX = column)
                                <1>       ;  EDX = size (rows, colums)
                                <1>       ;         (HW = rows, DX = columns)
                                <1>       ;       (0 -> invalid
                                <1>       ;         (1 -> horizontal or vertical line)
                                <1>       ;  [maskcolor] = mask color (to be excluded)
                                <1>       ;
                                <1>       ; OUTPUT:
                                <1>       ;      [u.r0] will be > 0 if succesful
                                <1>
0000E978 F605[70120300]10       <1>       test   byte [v_ops], 10h ; display page or window ?
0000E97F 7553                   <1>       jnz    short pix_op_inc_w ; window
                                <1>
0000E981 8B3D[72120300]         <1>       mov    edi, [v_mem]
0000E987 89FE                   <1>       mov    esi, edi
0000E989 8B0D[76120300]         <1>       mov    ecx, [v_siz] ; display page pixel count
                                <1>
0000E98F F605[70120300]20       <1>       test   byte [v_ops], 20h ; masked increase color ?
0000E996 7405                   <1>       jz     short pix_op_inc_0 ; no
0000E998 E90C100000             <1>       jmp    m_pix_op_inc ; 'inc' color except mask color
                                <1> pix_op_inc_0:
0000E99D 803D[71120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000E9A4 7707                   <1>       ja     short pix_op_inc_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0000E9A6 E8EB040000             <1>       call   pix_op_inc_8
0000E9AB EB1E                   <1>       jmp    short pix_op_inc_4
                                <1>
                                <1> pix_op_inc_1:
0000E9AD 803D[71120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000E9B4 7710                   <1>       ja     short pix_op_inc_3 ; 32bpp
0000E9B6 7207                   <1>       jb     short pix_op_inc_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0000E9B8 E8F0040000             <1>       call   pix_op_inc_24
0000E9BD EB0C                   <1>       jmp    short pix_op_inc_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> pix_op_inc_2:
0000E9BF E8DC040000             <1>       call   pix_op_inc_16
0000E9C4 EB05                   <1>       jmp    short pix_op_inc_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> pix_op_inc_3:
0000E9C6 E8F6040000             <1>       call   pix_op_inc_32
                                <1> pix_op_inc_4:
0000E9CB 29F7                   <1>       sub    edi, esi
0000E9CD 893D[64030300]         <1>       mov    [u.r0], edi
                                <1> pix_op_inc_5:
0000E9D3 C3                     <1>       retn
                                <1>
                                <1> pix_op_inc_w:
                                <1>       ; 31/01/2021
                                <1>       ; ecx = win start pos (row, column)
                                <1>       ; edx = size (rows, columns)
0000E9D4 E834F7FFFF             <1>       call   sysvideo_15_12 ; window preparations
0000E9D9 72F8                   <1>       jc     short pix_op_inc_5
                                <1>
0000E9DB F605[70120300]20       <1>       test   byte [v_ops], 20h ; masked increase color ?
0000E9E2 7405                   <1>       jz     short pix_op_inc_w_0 ; no
0000E9E4 E959100000             <1>       jmp    m_pix_op_inc_w
                                <1>                    ; window 'inc' color except mask color
                                <1> pix_op_inc_w_0:
                                <1>       ; ecx = bytes per row (to be applied)
                                <1>       ; edx = screen width in bytes
                                <1>       ; ebx = row count
                                <1>
0000E9E9 8B3D[7A120300]         <1>       mov    edi, [v_str]
                                <1>
0000E9EF 803D[71120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000E9F6 7707                   <1>       ja     short pix_op_inc_w_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0000E9F8 BD[96EE0000]           <1>       mov    ebp, pix_op_inc_8
0000E9FD EB1E                   <1>       jmp    short pix_op_inc_w_4
                                <1>
                                <1> pix_op_inc_w_1:
0000E9FF 803D[71120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000EA06 7710                   <1>       ja     short pix_op_inc_w_3 ; 32bpp
0000EA08 7207                   <1>       jb     short pix_op_inc_w_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0000EA0A BD[ADEE0000]           <1>       mov    ebp, pix_op_inc_24
```

```
8719 0000EA0F EB0C                <1>       jmp    short pix_op_inc_w_4
8720                              <1>
8721                              <1>       ; 65536 colors (16bpp)
8722                              <1> pix_op_inc_w_2:
8723 0000EA11 BD[A0EE0000]        <1>       mov    ebp, pix_op_inc_16
8724 0000EA16 EB05                <1>       jmp    short pix_op_inc_w_4
8725                              <1>
8726                              <1>       ; 32 bit true colors
8727                              <1> pix_op_inc_w_3:
8728 0000EA18 BD[C1EE0000]        <1>       mov    ebp, pix_op_inc_32
8729                              <1> pix_op_inc_w_4:
8730 0000EA1D E9EAFDFFFF          <1>       jmp    pix_op_inc_w_x
8731                              <1>
8732                              <1> pix_op_dec:
8733                              <1>       ; 31/01/2021
8734                              <1>       ; DECREASE COLOR
8735                              <1>       ;
8736                              <1>       ; INPUT:
8737                              <1>       ;  ECX = start position (row, column)
8738                              <1>       ;         (HW = row, CX = column)
8739                              <1>       ;  EDX = size (rows, colums)
8740                              <1>       ;         (HW = rows, DX = columns)
8741                              <1>       ;       (0 -> invalid
8742                              <1>       ;         (1 -> horizontal or vertical line)
8743                              <1>       ;  [maskcolor] = mask color (to be excluded)
8744                              <1>       ;
8745                              <1>       ; OUTPUT:
8746                              <1>       ;      [u.r0] will be > 0 if succesful
8747                              <1>
8748 0000EA22 F605[70120300]10    <1>       test   byte [v_ops], 10h ; display page or window ?
8749 0000EA29 7553                <1>       jnz    short pix_op_dec_w ; window
8750                              <1>
8751 0000EA2B 8B3D[72120300]      <1>       mov    edi, [v_mem]
8752 0000EA31 89FE                <1>       mov    esi, edi
8753 0000EA33 8B0D[76120300]      <1>       mov    ecx, [v_siz] ; display page pixel count
8754                              <1>
8755 0000EA39 F605[70120300]20    <1>       test   byte [v_ops], 20h ; masked decrease color ?
8756 0000EA40 7405                <1>       jz     short pix_op_dec_0 ; no
8757 0000EA42 E92E100000          <1>       jmp    m_pix_op_dec ; 'dec' color except mask color
8758                              <1> pix_op_dec_0:
8759 0000EA47 803D[71120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
8760 0000EA4E 7707                <1>       ja     short pix_op_dec_1
8761                              <1>
8762                              <1>       ; 256 colors (8bpp)
8763 0000EA50 E878040000          <1>       call   pix_op_dec_8
8764 0000EA55 EB1E                <1>       jmp    short pix_op_dec_4
8765                              <1>
8766                              <1> pix_op_dec_1:
8767 0000EA57 803D[71120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
8768 0000EA5E 7710                <1>       ja     short pix_op_dec_3 ; 32bpp
8769 0000EA60 7207                <1>       jb     short pix_op_dec_2 ; 16bpp
8770                              <1>
8771                              <1>       ; 24 bit true colors
8772 0000EA62 E87D040000          <1>       call   pix_op_dec_24
8773 0000EA67 EB0C                <1>       jmp    short pix_op_dec_4
8774                              <1>
8775                              <1>       ; 65536 colors (16bpp)
8776                              <1> pix_op_dec_2:
8777 0000EA69 E869040000          <1>       call   pix_op_dec_16
8778 0000EA6E EB05                <1>       jmp    short pix_op_dec_4
8779                              <1>
8780                              <1>       ; 32 bit true colors
8781                              <1> pix_op_dec_3:
8782 0000EA70 E882040000          <1>       call   pix_op_dec_32
8783                              <1> pix_op_dec_4:
8784 0000EA75 29F7                <1>       sub    edi, esi
8785 0000EA77 893D[64030300]      <1>       mov    [u.r0], edi
8786                              <1> pix_op_dec_5:
8787 0000EA7D C3                  <1>       retn
8788                              <1>
8789                              <1> pix_op_dec_w:
8790                              <1>       ; 31/01/2021
8791                              <1>       ; ecx = win start pos (row, column)
8792                              <1>       ; edx = size (rows, columns)
8793 0000EA7E E88AF6FFFF          <1>       call   sysvideo_15_12 ; window preparations
8794 0000EA83 72F8                <1>       jc     short pix_op_dec_5
8795                              <1>
8796 0000EA85 F605[70120300]20    <1>       test   byte [v_ops], 20h ; masked decrease color ?
8797 0000EA8C 7405                <1>       jz     short pix_op_dec_w_0 ; no
8798 0000EA8E E976100000          <1>       jmp    m_pix_op_dec_w
8799                              <1>                      ; window 'dec' color except mask color
8800                              <1> pix_op_dec_w_0:
8801                              <1>       ; ecx = bytes per row (to be applied)
8802                              <1>       ; edx = screen width in bytes
8803                              <1>       ; ebx = row count
8804                              <1>
8805 0000EA93 8B3D[7A120300]      <1>       mov    edi, [v_str]
8806                              <1>
8807 0000EA99 803D[71120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
8808 0000EAA0 7707                <1>       ja     short pix_op_dec_w_1
8809                              <1>
8810                              <1>       ; 256 colors (8bpp)
8811 0000EAA2 BD[CDEE0000]        <1>       mov    ebp, pix_op_dec_8
8812 0000EAA7 EB1E                <1>       jmp    short pix_op_dec_w_4
8813                              <1>
8814                              <1> pix_op_dec_w_1:
8815 0000EAA9 803D[71120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
8816 0000EAB0 7710                <1>       ja     short pix_op_dec_w_3 ; 32bpp
8817 0000EAB2 7207                <1>       jb     short pix_op_dec_w_2 ; 16bpp
8818                              <1>
8819                              <1>       ; 24 bit true colors
8820 0000EAB4 BD[E4EE0000]        <1>       mov    ebp, pix_op_dec_24
8821 0000EAB9 EB0C                <1>       jmp    short pix_op_dec_w_4
8822                              <1>
8823                              <1>       ; 65536 colors (16bpp)
```

```
                            <1> pix_op_dec_w_2:
0000EABB BD[D7EE0000]       <1>         mov    ebp, pix_op_dec_16
0000EAC0 EB05               <1>         jmp    short pix_op_dec_w_4
                            <1>
                            <1>         ; 32 bit true colors
                            <1> pix_op_dec_w_3:
0000EAC2 BD[F7EE0000]       <1>         mov    ebp, pix_op_dec_32
                            <1> pix_op_dec_w_4:
0000EAC7 E940FDFFFF         <1>         jmp    pix_op_dec_w_x
                            <1>
                            <1> pix_op_blk:
                            <1>         ; 23/01/2021
                            <1>         ; 22/02/2021
                            <1>         ; 02/02/2021
                            <1>         ; COPY PIXEL BLOCK -system to system-
                            <1>         ; WRITE PIXEL BLOCKS -user to system-
                            <1>         ;
                            <1>         ; INPUT:
                            <1>         ;   -If BL bit 5 is 0-
                            <1>         ;    ECX = start position (row, column) (*)
                            <1>         ;    (HW = row, CX = column)
                            <1>         ;    EDX = size (rows, colums) (*)
                            <1>         ;    (HW = rows, DX = columns)
                            <1>         ;         (0 -> invalid)
                            <1>         ;         (1 -> horizontal or vertical line)
                            <1>         ;    ESI = destination (row, column) (***)
                            <1>         ;   -If BL bit 5 is 1-
                            <1>         ;     CL = color (8 bit, 256 colors)
                            <1>         ;    ECX = color (16 bit and true colors)
                            <1>         ;    EDX = count of blocks (not bytes)
                            <1>         ;       (limit: 2048 blocks/windows)
                            <1>         ;    ESI = user's buffer address
                            <1>         ;        contains 64 bit block data
                            <1>         ;        BLOCK ADDRESS - (row, col), dword
                            <1>         ;        (first 32 bits)
                            <1>         ;        BLOCK SIZE - (rows, cols), dword
                            <1>         ;        (second 32 bits)
                            <1>         ; OUTPUT:
                            <1>         ;     [u.r0] will be > 0 if succesful
                            <1>
                            <1>         ; Window option ([v_ops] bit 4) will be ignored
                            <1>         ; (Function is used for display page coordinates)
                            <1>
0000EACC F605[70120300]20   <1>         test   byte [v_ops], 20h ; masked or direct ?
0000EAD3 755A               <1>         jnz    short pix_op_blk_u ; blocks from user's buffer
                            <1>
0000EAD5 89F0               <1>         mov    eax, esi ; destination position (row, col)
0000EAD7 E8DEF6FFFF         <1>         call   calc_pixel_offset
0000EADC 3B05[76120300]     <1>         cmp    eax, [v_siz]
0000EAE2 734A               <1>         jnb    short pix_op_blk_retn ; out of display page
0000EAE4 89C6               <1>         mov    esi, eax
0000EAE6 E8ACF6FFFF         <1>         call   pixels_to_byte_count
0000EAEB 89C7               <1>         mov    edi, eax
0000EAED 89D0               <1>         mov    eax, edx  ; size
0000EAEF E8C6F6FFFF         <1>         call   calc_pixel_offset
                            <1>         ; 22/02/2021
0000EAF4 3B05[76120300]     <1>         cmp    eax, [v_siz]
0000EAFA 7732               <1>         ja     short pix_op_blk_retn ; out of display page
0000EAFC 01C6               <1>         add    esi, eax
0000EAFE 3B35[76120300]     <1>         cmp    esi, [v_siz]
0000EB04 7728               <1>         ja     short pix_op_blk_retn ; out of display page
                            <1>
0000EB06 033D[72120300]     <1>         add    edi, [v_mem] ; destination address
                            <1>
                            <1>         ; 23/01/2021
                            <1>         ;call  pixels_to_byte_count
                            <1>         ;add   edi, eax
                            <1>         ;jc    short pix_op_blk_retn ; out of display page
                            <1>         ;cmp   edi, [v_end]
                            <1>         ;ja    short pix_op_blk_retn ; out of display page
                            <1>         ;sub   edi, eax
                            <1>
0000EB0C E8FCF5FFFF         <1>         call   sysvideo_15_12 ; window preparations
0000EB11 721B               <1>         jc     short pix_op_blk_retn ; something wrong !?
                            <1>         ; ecx = bytes per row (to be applied)
                            <1>         ; edx = screen width in bytes
                            <1>         ; ebx = row count
                            <1>
0000EB13 8B35[7A120300]     <1>         mov    esi, [v_str] ; source address
                            <1>
                            <1>         ; Note:
                            <1>         ; ecx & edx are already adjusted for pixel sizes
                            <1>         ; so, following code is proper all pixel sizes
                            <1>
0000EB19 29CA               <1>         sub    edx, ecx ; screen width - window width
                            <1> pix_op_blk_0:
0000EB1B 89C8               <1>         mov    eax, ecx
0000EB1D 0105[64030300]     <1>         add    [u.r0], eax
0000EB23 F3A4               <1>         rep    movsb
0000EB25 89C1               <1>         mov    ecx, eax
0000EB27 01D6               <1>         add    esi, edx ; next row
0000EB29 01D7               <1>         add    edi, edx ; next row
0000EB2B 4B                 <1>         dec    ebx
0000EB2C 75ED               <1>         jnz    short pix_op_blk_0
                            <1> pix_op_blk_retn:
0000EB2E C3                 <1>         retn
                            <1>
                            <1> pix_op_blk_u:
                            <1>         ; fill blocks (windows) with desired color
                            <1>         ; according to block definitions in user's buffer
0000EB2F 81FA00080000       <1>         cmp    edx, 2048
0000EB35 7605               <1>         jna    short pix_op_blk_u_0
                            <1>         ; Maximum 2048 blocks
0000EB37 BA00080000         <1>         mov    edx, 2048
                            <1> pix_op_blk_u_0:
```

```
8929 0000EB3C 8025[70120300]DF    <1>       and   byte [v_ops], ~20h ; clear masked bit
8930 0000EB43 890D[82120300]      <1>       mov   [maskcolor], ecx ; save pixel color
8931                              <1>       ; 22/02/2021
8932                              <1>       ;mov  ebp, edx ; save blocks count
8933                              <1>       ;push ebp
8934                              <1> pix_op_blk_u_next:
8935 0000EB49 52                 <1>       push  edx
8936 0000EB4A B908000000          <1>       mov   ecx, 8
8937 0000EB4F BF[8A120300]        <1>       mov   edi, buffer8 ; 8 bytes small buffer
8938                              <1>       ; esi = user's buffer address
8939 0000EB54 E8332A0000          <1>       call  transfer_from_user_buffer
8940 0000EB59 72D3                <1>       jc    short pix_op_blk_retn
8941 0000EB5B 01CE                <1>       add   esi, ecx ; 22/02/2021
8942 0000EB5D 56                  <1>       push  esi
8943 0000EB5E 8B15[8A120300]      <1>       mov   edx, [buffer8] ; block start pos (row,col)
8944 0000EB64 8B35[8E120300]      <1>       mov   esi, [buffer8+4] ; block size (rows,cols)
8945 0000EB6A 8B0D[82120300]      <1>       mov   ecx, [maskcolor]
8946 0000EB70 E848FCFFFF          <1>       call  pix_op_new_w ; new (change) color (window)
8947 0000EB75 5E                  <1>       pop   esi
8948                              <1>       ;pop  ebp
8949                              <1>       ;dec  ebp
8950 0000EB76 5A                  <1>       pop   edx
8951 0000EB77 4A                  <1>       dec   edx
8952 0000EB78 75CF                <1>       jnz   short pix_op_blk_u_next
8953 0000EB7A C3                  <1>       retn
8954                              <1>
8955                              <1> pix_op_lin:
8956                              <1>       ; 12/02/2021
8957                              <1>       ; 11/02/2021
8958                              <1>       ; 10/02/2021
8959                              <1>       ; 05/02/2021
8960                              <1>       ; 02/02/2021
8961                              <1>       ; WRITE LINE -direct-
8962                              <1>       ; WRITE LINE(S) -via user's buffer-
8963                              <1>       ;
8964                              <1>       ; INPUT:
8965                              <1>       ;   -If BL bit 5 is 0-
8966                              <1>       ;     CL = color (8 bit, 256 colors)
8967                              <1>       ;    ECX = color (16 bit and true colors)
8968                              <1>       ;     DX = low 12 bits - size (length)
8969                              <1>       ;         high 4 bits - direction or type
8970                              <1>       ;           0 - Horizontal line
8971                              <1>       ;           1 - Vertical line
8972                              <1>       ;         > 1 - undefined, invalid
8973                              <1>       ;    ESI = start position (row, column)
8974                              <1>       ;        (HW = row, SI = column)
8975                              <1>       ;   -If BL bit 5 is 1-
8976                              <1>       ;     CL = color (8 bit, 256 colors)
8977                              <1>       ;    ECX = color (16 bit and true colors)
8978                              <1>       ;     DX = number of lines (in user buffer)
8979                              <1>       ;         (limit: 2048 lines)
8980                              <1>       ;    ESI = user's buffer
8981                              <1>       ;          contains 64 bit data for lines
8982                              <1>       ;         START POINT: 32 bit (row, col)
8983                              <1>       ;         LENGTH: 32 bit
8984                              <1>       ;               high 16 bits - 0
8985                              <1>       ;               bit 0-11 - length
8986                              <1>       ;               bit 12-15 - type (length)
8987                              <1>       ; OUTPUT:
8988                              <1>       ;     [u.r0] will be > 0 if succesful
8989                              <1>
8990                              <1>       ; Window option ([v_ops] bit 4) will be ignored
8991                              <1>       ; (Function is used for display page coordinates)
8992                              <1>
8993                              <1>       ; 10/02/2021
8994 0000EB7B F605[70120300]20    <1>       test  byte [v_ops], 20h ; masked or direct ?
8995 0000EB82 7445                <1>       jz    short pix_op_lin_vh ; direct (v/h lines)
8996                              <1>
8997                              <1>       ; lines from user's buffer
8998                              <1> pix_op_lin_u:
8999                              <1>       ; draw lines with desired color
9000                              <1>       ; according to line definitions in user's buffer
9001 0000EB84 81FA00080000        <1>       cmp   edx, 2048
9002 0000EB8A 7605                <1>       jna   short pix_op_lin_u_0
9003                              <1>       ; Maximum 2048 lines
9004 0000EB8C BA00080000          <1>       mov   edx, 2048
9005                              <1> pix_op_lin_u_0:
9006 0000EB91 890D[82120300]      <1>       mov   [maskcolor], ecx ; save pixel color
9007 0000EB97 89D5                <1>       mov   ebp, edx ; save line count
9008                              <1> pix_op_lin_u_next:
9009 0000EB99 B908000000          <1>       mov   ecx, 8
9010 0000EB9E BF[8A120300]        <1>       mov   edi, buffer8 ; 8 bytes small buffer
9011                              <1>       ; esi = user's buffer address
9012 0000EBA3 E8E4290000          <1>       call  transfer_from_user_buffer
9013 0000EBA8 721E                <1>       jc    short pix_op_lin_retn
9014 0000EBAA 01CE                <1>       add   esi, ecx ; 11/02/2021
9015 0000EBAC 56                  <1>       push  esi
9016 0000EBAD 8B35[8A120300]      <1>       mov   esi, [buffer8] ; line start pos (row,col)
9017 0000EBB3 8B15[8E120300]      <1>       mov   edx, [buffer8+4] ; line length
9018 0000EBB9 8B0D[82120300]      <1>       mov   ecx, [maskcolor]
9019 0000EBBF E805000000          <1>       call  pix_op_lin_vh ; new (change) color (window)
9020 0000EBC4 5E                  <1>       pop   esi
9021 0000EBC5 4D                  <1>       dec   ebp
9022 0000EBC6 75D1                <1>       jnz   short pix_op_lin_u_next
9023                              <1> pix_op_lin_retn:
9024 0000EBC8 C3                  <1>       retn
9025                              <1>
9026                              <1> pix_op_lin_vh:
9027 0000EBC9 81FA38140000        <1>       cmp   edx, 1438h ; 1920*1080 (780hx438h) limit
9028 0000EBCF 7761                <1>       ja    short pix_op_lin_err1 ; invalid type
9029                              <1>                           ; (for current version)
9030 0000EBD1 66F7C2FF0F          <1>       test  dx, 0FFFh
9031 0000EBD6 745A                <1>       jz    short pix_op_lin_err1 ; zero length!
9032                              <1>
9033 0000EBD8 89F0                <1>       mov   eax, esi ; start point (row, col)
```

```
9034 0000EBDA E8DBF5FFFF          <1>       call   calc_pixel_offset
9035 0000EBDF 3B05[76120300]      <1>       cmp    eax, [v_siz]
9036 0000EBE5 734B                <1>       jnb    short pix_op_lin_err1 ; out of display page!
9037 0000EBE7 E8ABF5FFFF          <1>       call   pixels_to_byte_count
9038 0000EBEC 89C7                <1>       mov    edi, eax ; start point offset
9039 0000EBEE 033D[72120300]      <1>       add    edi, [v_mem] ; LFB start address
9040 0000EBF4 89C8                <1>       mov    eax, ecx ; color
9041                              <1>
9042 0000EBF6 F6C610              <1>       test   dh, 10h
9043 0000EBF9 0F848A000000        <1>       jz     pix_op_lin_h ; Horizontal line
9044                              <1>
9045                              <1> pix_op_lin_v:
9046                              <1>       ; Vertical line
9047 0000EBFF 80E60F              <1>       and    dh, 0Fh ; low 12 bits
9048 0000EC02 51                  <1>       push   ecx ; color
9049 0000EC03 89D1                <1>       mov    ecx, edx
9050 0000EC05 0FB705[6E120300]    <1>       movzx  eax, word [v_width]
9051 0000EC0C 89C3                <1>       mov    ebx, eax
9052                              <1>       ; 12/02/2021
9053 0000EC0E F7E2                <1>       mul    edx ; rows * [v_width]
9054 0000EC10 01F8                <1>       add    eax, edi
9055 0000EC12 3B05[7E120300]      <1>       cmp    eax, [v_end]
9056 0000EC18 58                  <1>       pop    eax ; color
9057 0000EC19 7717                <1>       ja     short pix_op_lin_err1 ; out of display page
9058                              <1>       ; ecx = rows
9059 0000EC1B 89CA                <1>       mov    edx, ecx
9060                              <1>
9061 0000EC1D 803D[71120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
9062 0000EC24 770D                <1>       ja     short pix_op_lin_v_2
9063                              <1>       ; 256 colors (1 byte per pixel)
9064 0000EC26 010D[64030300]      <1>       add    [u.r0], ecx ; byte count
9065                              <1> pix_op_lin_v_1:
9066 0000EC2C 8807                <1>       mov    [edi], al
9067 0000EC2E 01DF                <1>       add    edi, ebx  ; next row
9068 0000EC30 E2FA                <1>       loop   pix_op_lin_v_1
9069                              <1> pix_op_lin_err1:
9070 0000EC32 C3                  <1>       retn
9071                              <1>
9072                              <1> pix_op_lin_v_2:
9073 0000EC33 803D[71120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
9074 0000EC3A 773A                <1>       ja     short pix_op_lin_v_6 ; 32bpp
9075 0000EC3C 7226                <1>       jb     short pix_op_lin_v_4 ; 16bpp
9076                              <1>
9077                              <1>       ; 24 bit true colors
9078                              <1>       ; * 3
9079 0000EC3E 53                  <1>       push   ebx ; screen width in pixels
9080 0000EC3F D1E3                <1>       shl    ebx, 1
9081 0000EC41 011C24              <1>       add    [esp], ebx
9082 0000EC44 5B                  <1>       pop    ebx ; screen width in bytes
9083 0000EC45 010D[64030300]      <1>       add    [u.r0], ecx
9084 0000EC4B D1E2                <1>       shl    edx, 1
9085 0000EC4D 0115[64030300]      <1>       add    [u.r0], edx ; byte count
9086                              <1> pix_op_lin_v_3:
9087 0000EC53 668907              <1>       mov    [edi], ax
9088 0000EC56 C1C810              <1>       ror    eax, 16
9089 0000EC59 884702              <1>       mov    [edi+2], al
9090 0000EC5C C1C010              <1>       rol    eax, 16
9091 0000EC5F 01DF                <1>       add    edi, ebx  ; next row
9092 0000EC61 E2F0                <1>       loop   pix_op_lin_v_3
9093 0000EC63 C3                  <1>       retn
9094                              <1>
9095                              <1> pix_op_lin_v_4:
9096                              <1>       ; 16 bit (65536) colors
9097 0000EC64 D1E3                <1>       shl    ebx, 1
9098 0000EC66 D1E2                <1>       shl    edx, 1
9099 0000EC68 0115[64030300]      <1>       add    [u.r0], edx
9100                              <1> pix_op_lin_v_5:
9101 0000EC6E 668907              <1>       mov    [edi], ax
9102 0000EC71 01DF                <1>       add    edi, ebx  ; next row
9103 0000EC73 E2F9                <1>       loop   pix_op_lin_v_5
9104 0000EC75 C3                  <1>       retn
9105                              <1>
9106                              <1> pix_op_lin_v_6:
9107                              <1>       ; 32 bit true colors
9108 0000EC76 C1E302              <1>       shl    ebx, 2
9109 0000EC79 C1E202              <1>       shl    edx, 2
9110 0000EC7C 0115[64030300]      <1>       add    [u.r0], edx ; byte count
9111                              <1> pix_op_lin_v_7:
9112 0000EC82 8907                <1>       mov    [edi], eax
9113 0000EC84 01DF                <1>       add    edi, ebx  ; next row
9114 0000EC86 E2FA                <1>       loop   pix_op_lin_v_7
9115 0000EC88 C3                  <1>       retn
9116                              <1>
9117                              <1> pix_op_lin_h:
9118                              <1>       ; Horizontal line
9119 0000EC89 80E60F              <1>       and    dh, 0Fh ; low 12 bits
9120 0000EC8C 89D1                <1>       mov    ecx, edx
9121 0000EC8E 6601D6              <1>       add    si, dx ; start column + columns
9122 0000EC91 663B35[6E120300]    <1>       cmp    si, [v_width] ; screen width
9123 0000EC98 7711                <1>       ja     short pix_op_lin_err2 ; out of columns limit
9124                              <1>
9125 0000EC9A 803D[71120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
9126 0000ECA1 7709                <1>       ja     short pix_op_lin_h_1
9127                              <1>       ; 256 colors (1 byte per pixel)
9128 0000ECA3 010D[64030300]      <1>       add    [u.r0], ecx
9129 0000ECA9 F3AA                <1>       rep    stosb
9130                              <1> pix_op_lin_err2:
9131 0000ECAB C3                  <1>       retn
9132                              <1>
9133                              <1> pix_op_lin_h_1:
9134 0000ECAC 803D[71120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
9135 0000ECB3 7728                <1>       ja     short pix_op_lin_h_4 ; 32bpp
9136 0000ECB5 721A                <1>       jb     short pix_op_lin_h_3 ; 16bpp
9137                              <1>
9138                              <1>       ; 24 bit true colors
```

```
                                <1>       ; * 3
0000ECB7 0115[64030300]         <1>       add    [u.r0], edx
0000ECBD D1E2                   <1>       shl    edx, 1
0000ECBF 0115[64030300]         <1>       add    [u.r0], edx
                                <1> pix_op_lin_h_2:
0000ECC5 66AB                   <1>       stosw
0000ECC7 C1C810                 <1>       ror    eax, 16
0000ECCA AA                     <1>       stosb
0000ECCB C1C010                 <1>       rol    eax, 16
0000ECCE E2F5                   <1>       loop   pix_op_lin_h_2
0000ECD0 C3                     <1>       retn
                                <1>
                                <1> pix_op_lin_h_3:
                                <1>       ; 16 bit (65536) colors
0000ECD1 D1E2                   <1>       shl    edx, 1
0000ECD3 0115[64030300]         <1>       add    [u.r0], edx
0000ECD9 F366AB                 <1>       rep    stosw
0000ECDC C3                     <1>       retn
                                <1>
                                <1> pix_op_lin_h_4:
                                <1>       ; 32 bit true colors
0000ECDD C1E202                 <1>       shl    edx, 2
0000ECE0 0115[64030300]         <1>       add    [u.r0], edx
0000ECE6 F3AB                   <1>       rep    stosd
0000ECE8 C3                     <1>       retn
                                <1>
                                <1> pix_op_new_8:
                                <1>       ; 8 bit colors (256 colors)
                                <1>       ; CHANGE PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ;  al = color
                                <1>       ; edi = start pixel address
                                <1>
0000ECE9 F3AA                   <1>       rep    stosb
0000ECEB C3                     <1>       retn
                                <1>
                                <1> pix_op_new_16:
                                <1>       ; 16 bit colors (65536 colors)
                                <1>       ; CHANGE PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ;  ax = color
                                <1>       ; edi = start pixel address
                                <1>
0000ECEC F366AB                 <1>       rep    stosw
0000ECEF C3                     <1>       retn
                                <1>
                                <1> pix_op_new_24:
                                <1>       ; 24 bit true colors
                                <1>       ; CHANGE PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ; eax = color
                                <1>       ; edi = start pixel address
                                <1>
0000ECF0 66AB                   <1>       stosw
0000ECF2 C1C810                 <1>       ror    eax, 16
0000ECF5 AA                     <1>       stosb
0000ECF6 C1C010                 <1>       rol    eax, 16
0000ECF9 E2F5                   <1>       loop   pix_op_new_24
0000ECFB C3                     <1>       retn
                                <1>
                                <1> pix_op_new_32:
                                <1>       ; 32 bit true colors
                                <1>       ; CHANGE PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ; eax = color
                                <1>       ; edi = start pixel address
                                <1>
0000ECFC F3AB                   <1>       rep    stosd
0000ECFE C3                     <1>       retn
                                <1>
                                <1> pix_op_add_8:
                                <1>       ; 8 bit colors (256 colors)
                                <1>       ; ADD PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ;  al = color
                                <1>       ; edi = start pixel address
                                <1>
0000ECFF 88C4                   <1>       mov    ah, al
                                <1> pix_op_add_8_0:
0000ED01 0207                   <1>       add    al, [edi]
0000ED03 7302                   <1>       jnc    short pix_op_add_8_1
0000ED05 B0FF                   <1>       mov    al, 0FFh ; Max. value
                                <1> pix_op_add_8_1:
0000ED07 AA                     <1>       stosb
0000ED08 88E0                   <1>       mov    al, ah
0000ED0A E2F5                   <1>       loop   pix_op_add_8_0
0000ED0C C3                     <1>       retn
                                <1>
                                <1> pix_op_add_16:
                                <1>       ; 16 bit colors (65536 colors)
                                <1>       ; ADD PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ;  ax = color
                                <1>       ; edi = start pixel address
                                <1>
0000ED0D 89C2                   <1>       mov    edx, eax
                                <1> pix_op_add_16_0:
0000ED0F 660307                 <1>       add    ax, [edi]
0000ED12 7304                   <1>       jnc    short pix_op_add_16_1
0000ED14 66B8FFFF               <1>       mov    ax, 0FFFFh ; Max. value
                                <1> pix_op_add_16_1:
0000ED18 66AB                   <1>       stosw
0000ED1A 89D0                   <1>       mov    eax, edx
0000ED1C E2F1                   <1>       loop   pix_op_add_16_0
0000ED1E C3                     <1>       retn
```

```
                                <1>
                                <1> pix_op_add_24:
                                <1>       ; 24 bit true colors
                                <1>       ; ADD PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ; eax = color
                                <1>       ; edi = start pixel address
                                <1>
0000ED1F 53                     <1>       push   ebx
0000ED20 BBFFFFFF00             <1>       mov    ebx, 0FFFFFFh
                                <1>       ;and   eax, ebx ; 0FFFFFFh
0000ED25 89C2                   <1>       mov    edx, eax
                                <1> pix_op_add_24_0:
0000ED27 8B07                   <1>       mov    eax, [edi]
0000ED29 21D8                   <1>       and    eax, ebx ; 0FFFFFFh
0000ED2B 01D0                   <1>       add    eax, edx
0000ED2D 39D8                   <1>       cmp    eax, ebx
0000ED2F 7602                   <1>       jna    short pix_op_add_24_1
0000ED31 89D8                   <1>       mov    eax, ebx ; 0FFFFFFh  ; Max. value
                                <1> pix_op_add_24_1:
0000ED33 66AB                   <1>       stosw
0000ED35 C1E810                 <1>       shr    eax, 16
0000ED38 AA                     <1>       stosb
0000ED39 E2EC                   <1>       loop   pix_op_add_24_0
0000ED3B 89D0                   <1>       mov    eax, edx
0000ED3D 5B                     <1>       pop    ebx
0000ED3E C3                     <1>       retn
                                <1>
                                <1> pix_op_add_32:
                                <1>       ; 32 bit true colors
                                <1>       ; ADD PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ; eax = color
                                <1>       ; edi = start pixel address
                                <1>
0000ED3F 89C2                   <1>       mov    edx, eax
                                <1> pix_op_add_32_0:
0000ED41 0307                   <1>       add    eax, [edi]
0000ED43 7303                   <1>       jnc    short pix_op_add_32_1
                                <1>       ;mov   eax, 0FFFFFFFFh ; Max. value
0000ED45 29C0                   <1>       sub    eax, eax
0000ED47 48                     <1>       dec    eax
                                <1> pix_op_add_32_1:
0000ED48 AB                     <1>       stosd
0000ED49 89D0                   <1>       mov    eax, edx
0000ED4B E2F4                   <1>       loop   pix_op_add_32_0
0000ED4D C3                     <1>       retn
                                <1>
                                <1> pix_op_sub_8:
                                <1>       ; 8 bit colors (256 colors)
                                <1>       ; SUBTRACT PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ;  al = color
                                <1>       ; edi = start pixel address
                                <1>
0000ED4E 88C4                   <1>       mov    ah, al
                                <1> pix_op_sub_8_0:
0000ED50 8A07                   <1>       mov    al, [edi]
0000ED52 28E0                   <1>       sub    al, ah
0000ED54 7302                   <1>       jnb    short pix_op_sub_8_1
0000ED56 30C0                   <1>       xor    al, al ; 0 ; Min. value
                                <1> pix_op_sub_8_1:
0000ED58 AA                     <1>       stosb
0000ED59 E2F5                   <1>       loop   pix_op_sub_8_0
0000ED5B 88E0                   <1>       mov    al, ah
0000ED5D C3                     <1>       retn
                                <1>
                                <1> pix_op_sub_16:
                                <1>       ; 16 bit colors (65536 colors)
                                <1>       ; SUBTRACT PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ;  ax = color
                                <1>       ; edi = start pixel address
                                <1>
0000ED5E 89C2                   <1>       mov    edx, eax
                                <1> pix_op_sub_16_0:
0000ED60 668B07                 <1>       mov    ax, [edi]
0000ED63 6629D0                 <1>       sub    ax, dx
0000ED66 7302                   <1>       jnb    short pix_op_sub_16_1
0000ED68 31C0                   <1>       xor    eax, eax ; 0 ; Min. value
                                <1> pix_op_sub_16_1:
0000ED6A 66AB                   <1>       stosw
0000ED6C E2F2                   <1>       loop   pix_op_sub_16_0
0000ED6E 89D0                   <1>       mov    eax, edx
0000ED70 C3                     <1>       retn
                                <1>
                                <1> pix_op_sub_24:
                                <1>       ; 24 bit true colors
                                <1>       ; SUBTRACT PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ; eax = color
                                <1>       ; edi = start pixel address
                                <1>
                                <1>       ;and   eax, 0FFFFFFh
0000ED71 89C2                   <1>       mov    edx, eax
                                <1> pix_op_sub_24_0:
0000ED73 8B07                   <1>       mov    eax, [edi]
                                <1>       ; 27/02/2021
0000ED75 25FFFFFF00             <1>       and    eax, 0FFFFFFh
0000ED7A 29D0                   <1>       sub    eax, edx
0000ED7C 7302                   <1>       jnb    short pix_op_sub_24_1
0000ED7E 31C0                   <1>       xor    eax, eax ; 0 ; Min. value
                                <1> pix_op_sub_24_1:
0000ED80 66AB                   <1>       stosw
0000ED82 C1E810                 <1>       shr    eax, 16
```

```
9349 0000ED85 AA                <1>       stosb
9350 0000ED86 E2EB              <1>       loop   pix_op_sub_24_0
9351 0000ED88 89D0              <1>       mov    eax, edx
9352 0000ED8A C3                <1>       retn
9353                            <1>
9354                            <1> pix_op_sub_32:
9355                            <1>       ; 32 bit true colors
9356                            <1>       ; SUBTRACT PIXEL COLOR
9357                            <1>       ; ecx = pixel count per row
9358                            <1>       ; eax = color
9359                            <1>       ; edi = start pixel address
9360                            <1>
9361 0000ED8B 89C2              <1>       mov    edx, eax
9362                            <1> pix_op_sub_32_0:
9363 0000ED8D 8B07              <1>       mov    eax, [edi]
9364 0000ED8F 29D0              <1>       sub    eax, edx
9365 0000ED91 7302              <1>       jnb    short pix_op_sub_32_1
9366 0000ED93 31C0              <1>       xor    eax, eax ; 0 ; Min. value
9367                            <1> pix_op_sub_32_1:
9368 0000ED95 AB                <1>       stosd
9369 0000ED96 E2F5              <1>       loop   pix_op_sub_32_0
9370 0000ED98 89D0              <1>       mov    eax, edx
9371 0000ED9A C3                <1>       retn
9372                            <1>
9373                            <1> pix_op_or_8:
9374                            <1>       ; 8 bit colors (256 colors)
9375                            <1>       ; OR PIXEL COLOR
9376                            <1>       ; ecx = pixel count per row
9377                            <1>       ;  al = color
9378                            <1>       ; edi = start pixel address
9379                            <1>
9380                            <1> pix_op_or_8_0:
9381 0000ED9B 0807              <1>       or     [edi], al
9382 0000ED9D 47                <1>       inc    edi
9383 0000ED9E E2FB              <1>       loop   pix_op_or_8_0
9384 0000EDA0 C3                <1>       retn
9385                            <1>
9386                            <1> pix_op_or_16:
9387                            <1>       ; 16 bit colors (65536 colors)
9388                            <1>       ; OR PIXEL COLOR
9389                            <1>       ; ecx = pixel count per row
9390                            <1>       ;  ax = color
9391                            <1>       ; edi = start pixel address
9392                            <1>
9393                            <1> pix_op_or_16_0:
9394 0000EDA1 660907            <1>       or     [edi], ax
9395 0000EDA4 47                <1>       inc    edi
9396 0000EDA5 47                <1>       inc    edi
9397 0000EDA6 E2F9              <1>       loop   pix_op_or_16_0
9398 0000EDA8 C3                <1>       retn
9399                            <1>
9400                            <1> pix_op_or_24:
9401                            <1>       ; 24 bit true colors
9402                            <1>       ; OR PIXEL COLOR
9403                            <1>       ; ecx = pixel count per row
9404                            <1>       ; eax = color
9405                            <1>       ; edi = start pixel address
9406                            <1>
9407 0000EDA9 89C2              <1>       mov    edx, eax
9408                            <1> pix_op_or_24_0:
9409 0000EDAB 0B07              <1>       or     eax, [edi]
9410 0000EDAD 66AB              <1>       stosw
9411 0000EDAF C1E810            <1>       shr    eax, 16
9412 0000EDB2 AA                <1>       stosb
9413 0000EDB3 89D0              <1>       mov    eax, edx
9414 0000EDB5 E2F4              <1>       loop   pix_op_or_24_0
9415 0000EDB7 C3                <1>       retn
9416                            <1>
9417                            <1> pix_op_or_32:
9418                            <1>       ; 32 bit true colors
9419                            <1>       ; OR PIXEL COLOR
9420                            <1>       ; ecx = pixel count per row
9421                            <1>       ; eax = color
9422                            <1>       ; edi = start pixel address
9423                            <1>
9424                            <1>       ;mov   edx, eax
9425                            <1> pix_op_or_32_0:
9426                            <1>       ;or    eax, [edi]
9427                            <1>       ;stosd
9428                            <1>       ;mov   eax, edx
9429 0000EDB8 0907              <1>       or     [edi], eax
9430 0000EDBA 83C704            <1>       add    edi, 4
9431 0000EDBD E2F9              <1>       loop   pix_op_or_32_0
9432 0000EDBF C3                <1>       retn
9433                            <1>
9434                            <1> pix_op_and_8:
9435                            <1>       ; 8 bit colors (256 colors)
9436                            <1>       ; AND PIXEL COLOR
9437                            <1>       ; ecx = pixel count per row
9438                            <1>       ;  al = color
9439                            <1>       ; edi = start pixel address
9440                            <1>
9441                            <1> pix_op_and_8_0:
9442 0000EDC0 2007              <1>       and    [edi], al
9443 0000EDC2 47                <1>       inc    edi
9444 0000EDC3 E2FB              <1>       loop   pix_op_and_8_0
9445 0000EDC5 C3                <1>       retn
9446                            <1>
9447                            <1> pix_op_and_16:
9448                            <1>       ; 16 bit colors (65536 colors)
9449                            <1>       ; AND PIXEL COLOR
9450                            <1>       ; ecx = pixel count per row
9451                            <1>       ;  ax = color
9452                            <1>       ; edi = start pixel address
9453                            <1>
```

```
9454                                    <1> pix_op_and_16_0:
9455 0000EDC6 662107                    <1>         and   [edi], ax
9456 0000EDC9 47                        <1>         inc   edi
9457 0000EDCA 47                        <1>         inc   edi
9458 0000EDCB E2F9                      <1>         loop  pix_op_and_16_0
9459 0000EDCD C3                        <1>         retn
9460                                    <1>
9461                                    <1> pix_op_and_24:
9462                                    <1>         ; 24 bit true colors
9463                                    <1>         ; AND PIXEL COLOR
9464                                    <1>         ; ecx = pixel count per row
9465                                    <1>         ; eax = color
9466                                    <1>         ; edi = start pixel address
9467                                    <1>
9468 0000EDCE 89C2                      <1>         mov   edx, eax
9469                                    <1> pix_op_and_24_0:
9470 0000EDD0 2307                      <1>         and   eax, [edi]
9471 0000EDD2 66AB                      <1>         stosw
9472 0000EDD4 C1E810                    <1>         shr   eax, 16
9473 0000EDD7 AA                        <1>         stosb
9474 0000EDD8 89D0                      <1>         mov   eax, edx
9475 0000EDDA E2F4                      <1>         loop  pix_op_and_24_0
9476 0000EDDC C3                        <1>         retn
9477                                    <1>
9478                                    <1> pix_op_and_32:
9479                                    <1>         ; 32 bit true colors
9480                                    <1>         ; AND PIXEL COLOR
9481                                    <1>         ; ecx = pixel count per row
9482                                    <1>         ; eax = color
9483                                    <1>         ; edi = start pixel address
9484                                    <1>
9485                                    <1>         ;mov   edx, eax
9486                                    <1> pix_op_and_32_0:
9487                                    <1>         ;and   eax, [edi]
9488                                    <1>         ;stosd
9489                                    <1>         ;mov   eax, edx
9490 0000EDDD 2107                      <1>         and   [edi], eax
9491 0000EDDF 83C704                    <1>         add   edi, 4
9492 0000EDE2 E2F9                      <1>         loop  pix_op_and_32_0
9493 0000EDE4 C3                        <1>         retn
9494                                    <1>
9495                                    <1> pix_op_xor_8:
9496                                    <1>         ; 8 bit colors (256 colors)
9497                                    <1>         ; XOR PIXEL COLOR
9498                                    <1>         ; ecx = pixel count per row
9499                                    <1>         ;  al = color
9500                                    <1>         ; edi = start pixel address
9501                                    <1>
9502                                    <1> pix_op_xor_8_0:
9503 0000EDE5 3007                      <1>         xor   [edi], al
9504 0000EDE7 47                        <1>         inc   edi
9505 0000EDE8 E2FB                      <1>         loop  pix_op_xor_8_0
9506 0000EDEA C3                        <1>         retn
9507                                    <1>
9508                                    <1> pix_op_xor_16:
9509                                    <1>         ; 16 bit colors (65536 colors)
9510                                    <1>         ; XOR PIXEL COLOR
9511                                    <1>         ; ecx = pixel count per row
9512                                    <1>         ;  ax = color
9513                                    <1>         ; edi = start pixel address
9514                                    <1>
9515                                    <1> pix_op_xor_16_0:
9516 0000EDEB 663107                    <1>         xor   [edi], ax
9517 0000EDEE 47                        <1>         inc   edi
9518 0000EDEF 47                        <1>         inc   edi
9519 0000EDF0 E2F9                      <1>         loop  pix_op_xor_16_0
9520 0000EDF2 C3                        <1>         retn
9521                                    <1>
9522                                    <1> pix_op_xor_24:
9523                                    <1>         ; 24 bit true colors
9524                                    <1>         ; XOR PIXEL COLOR
9525                                    <1>         ; ecx = pixel count per row
9526                                    <1>         ; eax = color
9527                                    <1>         ; edi = start pixel address
9528                                    <1>
9529 0000EDF3 89C2                      <1>         mov   edx, eax
9530                                    <1> pix_op_xor_24_0:
9531 0000EDF5 3307                      <1>         xor   eax, [edi]
9532 0000EDF7 66AB                      <1>         stosw
9533 0000EDF9 C1E810                    <1>         shr   eax, 16
9534 0000EDFC AA                        <1>         stosb
9535 0000EDFD 89D0                      <1>         mov   eax, edx
9536 0000EDFF E2F4                      <1>         loop  pix_op_xor_24_0
9537 0000EE01 C3                        <1>         retn
9538                                    <1>
9539                                    <1> pix_op_xor_32:
9540                                    <1>         ; 32 bit true colors
9541                                    <1>         ; XOR PIXEL COLOR
9542                                    <1>         ; ecx = pixel count per row
9543                                    <1>         ; eax = color
9544                                    <1>         ; edi = start pixel address
9545                                    <1>
9546                                    <1>         ;mov   edx, eax
9547                                    <1> pix_op_xor_32_0:
9548                                    <1>         ;xor   eax, [edi]
9549                                    <1>         ;stosd
9550                                    <1>         ;mov   eax, edx
9551 0000EE02 3107                      <1>         xor   [edi], eax
9552 0000EE04 83C704                    <1>         add   edi, 4
9553 0000EE07 E2F9                      <1>         loop  pix_op_xor_32_0
9554 0000EE09 C3                        <1>         retn
9555                                    <1>
9556                                    <1> pix_op_mix_8:
9557                                    <1>         ; 8 bit colors (256 colors)
9558                                    <1>         ; MIX (AVERAGE) PIXEL COLORS
```

```
9559                            <1>       ; ecx = pixel count per row
9560                            <1>       ;  al = color
9561                            <1>       ; edi = start pixel address
9562                            <1>
9563 0000EE0A 88C4              <1>       mov    ah, al
9564                            <1> pix_op_mix_8_0:
9565 0000EE0C 0207              <1>       add    al, [edi]
9566 0000EE0E D0D8              <1>       rcr    al, 1
9567 0000EE10 AA                <1>       stosb
9568 0000EE11 88E0              <1>       mov    al, ah
9569 0000EE13 E2F7              <1>       loop   pix_op_mix_8_0
9570 0000EE15 C3                <1>       retn
9571                            <1>
9572                            <1> pix_op_mix_16:
9573                            <1>       ; 16 bit colors (65536 colors)
9574                            <1>       ; MIX (AVERAGE) PIXEL COLORS
9575                            <1>       ; ecx = pixel count per row
9576                            <1>       ;  ax = color
9577                            <1>       ; edi = start pixel address
9578                            <1>
9579 0000EE16 89C2              <1>       mov    edx, eax
9580                            <1> pix_op_mix_16_0:
9581 0000EE18 660307            <1>       add    ax, [edi]
9582 0000EE1B 66D1D8            <1>       rcr    ax, 1
9583 0000EE1E 66AB              <1>       stosw
9584 0000EE20 89D0              <1>       mov    eax, edx
9585 0000EE22 E2F4              <1>       loop   pix_op_mix_16_0
9586 0000EE24 C3                <1>       retn
9587                            <1>
9588                            <1> pix_op_mix_24:
9589                            <1>       ; 24 bit true colors
9590                            <1>       ; MIX (AVERAGE) PIXEL COLORS
9591                            <1>       ; ecx = pixel count per row
9592                            <1>       ; eax = color
9593                            <1>       ; edi = start pixel address
9594                            <1>
9595 0000EE25 53                <1>       push   ebx
9596 0000EE26 BBFFFFFF00        <1>       mov    ebx, 0FFFFFFh
9597                            <1>       ;and   eax, ebx ; 0FFFFFFh
9598 0000EE2B 89C2              <1>       mov    edx, eax
9599                            <1> pix_op_mix_24_0:
9600 0000EE2D 8B07              <1>       mov    eax, [edi]
9601 0000EE2F 21D8              <1>       and    eax, ebx ; 0FFFFFFh
9602 0000EE31 01D0              <1>       add    eax, edx
9603 0000EE33 D1E8              <1>       shr    eax, 1
9604                            <1>       ;rcr   eax, 1
9605 0000EE35 66AB              <1>       stosw
9606 0000EE37 C1E810            <1>       shr    eax, 16
9607 0000EE3A AA                <1>       stosb
9608 0000EE3B E2F0              <1>       loop   pix_op_mix_24_0
9609 0000EE3D 89D0              <1>       mov    eax, edx
9610 0000EE3F 5B                <1>       pop    ebx
9611 0000EE40 C3                <1>       retn
9612                            <1>
9613                            <1> pix_op_mix_32:
9614                            <1>       ; 32 bit true colors
9615                            <1>       ; MIX (AVERAGE) PIXEL COLORS
9616                            <1>       ; ecx = pixel count per row
9617                            <1>       ; eax = color
9618                            <1>       ; edi = start pixel address
9619                            <1>
9620 0000EE41 89C2              <1>       mov    edx, eax
9621                            <1> pix_op_mix_32_0:
9622 0000EE43 0307              <1>       add    eax, [edi]
9623 0000EE45 D1D8              <1>       rcr    eax, 1
9624 0000EE47 AB                <1>       stosd
9625 0000EE48 89D0              <1>       mov    eax, edx
9626 0000EE4A E2F7              <1>       loop   pix_op_mix_32_0
9627 0000EE4C C3                <1>       retn
9628                            <1>
9629                            <1> pix_op_not_8:
9630                            <1>       ; 8 bit colors (256 colors)
9631                            <1>       ; NOT PIXEL COLOR
9632                            <1>       ; ecx = pixel count per row
9633                            <1>       ; edi = start pixel address
9634                            <1>
9635                            <1> pix_op_not_8_0:
9636 0000EE4D F617              <1>       not    byte [edi]
9637 0000EE4F 47                <1>       inc    edi
9638 0000EE50 E2FB              <1>       loop   pix_op_not_8_0
9639 0000EE52 C3                <1>       retn
9640                            <1>
9641                            <1> pix_op_not_16:
9642                            <1>       ; 16 bit colors (65536 colors)
9643                            <1>       ; NOT PIXEL COLOR
9644                            <1>       ; ecx = pixel count per row
9645                            <1>       ; edi = start pixel address
9646                            <1>
9647                            <1> pix_op_not_16_0:
9648 0000EE53 66F717            <1>       not    word [edi]
9649 0000EE56 47                <1>       inc    edi
9650 0000EE57 47                <1>       inc    edi
9651 0000EE58 E2F9              <1>       loop   pix_op_not_16_0
9652 0000EE5A C3                <1>       retn
9653                            <1>
9654                            <1> pix_op_not_24:
9655                            <1>       ; 24 bit true colors
9656                            <1>       ; NOT PIXEL COLOR
9657                            <1>       ; ecx = pixel count per row
9658                            <1>       ; edi = start pixel address
9659                            <1>
9660                            <1> pix_op_not_24_0:
9661 0000EE5B 66F717            <1>       not    word [edi]
9662 0000EE5E 47                <1>       inc    edi
9663 0000EE5F 47                <1>       inc    edi
```

```
9664 0000EE60 F617                <1>       not   byte [edi]
9665 0000EE62 47                  <1>       inc   edi
9666 0000EE63 E2F6                <1>       loop  pix_op_not_24_0
9667 0000EE65 C3                  <1>       retn
9668                              <1>
9669                              <1> pix_op_not_32:
9670                              <1>       ; 32 bit true colors
9671                              <1>       ; NOT PIXEL COLOR
9672                              <1>       ; ecx = pixel count per row
9673                              <1>       ; eax = color
9674                              <1>       ; edi = start pixel address
9675                              <1> pix_op_not_32_0:
9676 0000EE66 F717                <1>       not   dword [edi]
9677 0000EE68 83C704              <1>       add   edi, 4
9678 0000EE6B E2F9                <1>       loop  pix_op_not_32_0
9679 0000EE6D C3                  <1>       retn
9680                              <1>
9681                              <1> pix_op_neg_8:
9682                              <1>       ; 8 bit colors (256 colors)
9683                              <1>       ; NEG PIXEL COLOR
9684                              <1>       ; ecx = pixel count per row
9685                              <1>       ; edi = start pixel address
9686                              <1>
9687                              <1> pix_op_neg_8_0:
9688 0000EE6E F61F                <1>       neg   byte [edi]
9689 0000EE70 47                  <1>       inc   edi
9690 0000EE71 E2FB                <1>       loop  pix_op_neg_8_0
9691 0000EE73 C3                  <1>       retn
9692                              <1>
9693                              <1> pix_op_neg_16:
9694                              <1>       ; 16 bit colors (65536 colors)
9695                              <1>       ; NEG PIXEL COLOR
9696                              <1>       ; ecx = pixel count per row
9697                              <1>       ; edi = start pixel address
9698                              <1>
9699                              <1> pix_op_neg_16_0:
9700 0000EE74 66F71F              <1>       neg   word [edi]
9701 0000EE77 47                  <1>       inc   edi
9702 0000EE78 47                  <1>       inc   edi
9703 0000EE79 E2F9                <1>       loop  pix_op_neg_16_0
9704 0000EE7B C3                  <1>       retn
9705                              <1>
9706                              <1> pix_op_neg_24:
9707                              <1>       ; 24 bit true colors
9708                              <1>       ; NEG PIXEL COLOR
9709                              <1>       ; ecx = pixel count per row
9710                              <1>       ; edi = start pixel address
9711                              <1>
9712                              <1> pix_op_neg_24_0:
9713 0000EE7C 8B07                <1>       mov   eax, [edi]
9714 0000EE7E 25FFFFFF00          <1>       and   eax, 0FFFFFFh
9715 0000EE83 F7D8                <1>       neg   eax
9716 0000EE85 66AB                <1>       stosw
9717 0000EE87 C1E810              <1>       shr   eax, 16
9718 0000EE8A AA                  <1>       stosb
9719 0000EE8B E2EF                <1>       loop  pix_op_neg_24_0
9720 0000EE8D C3                  <1>       retn
9721                              <1>
9722                              <1> pix_op_neg_32:
9723                              <1>       ; 32 bit true colors
9724                              <1>       ; NEG PIXEL COLOR
9725                              <1>       ; ecx = pixel count per row
9726                              <1>       ; eax = color
9727                              <1>       ; edi = start pixel address
9728                              <1> pix_op_neg_32_0:
9729 0000EE8E F71F                <1>       neg   dword [edi]
9730 0000EE90 83C704              <1>       add   edi, 4
9731 0000EE93 E2F9                <1>       loop  pix_op_neg_32_0
9732 0000EE95 C3                  <1>       retn
9733                              <1>
9734                              <1> pix_op_inc_8:
9735                              <1>       ; 8 bit colors (256 colors)
9736                              <1>       ; INCREASE PIXEL COLOR
9737                              <1>       ; ecx = pixel count per row
9738                              <1>       ; edi = start pixel address
9739                              <1>
9740                              <1> pix_op_inc_8_0:
9741 0000EE96 FE07                <1>       inc   byte [edi]
9742 0000EE98 7502                <1>       jnz   short pix_op_inc_8_1
9743                              <1>       ;mov  [edi], 0FFh ; Max. value
9744 0000EE9A FE0F                <1>       dec   byte [edi]
9745                              <1> pix_op_inc_8_1:
9746 0000EE9C 47                  <1>       inc   edi
9747 0000EE9D E2F7                <1>       loop  pix_op_inc_8_0
9748 0000EE9F C3                  <1>       retn
9749                              <1>
9750                              <1> pix_op_inc_16:
9751                              <1>       ; 16 bit colors (65536 colors)
9752                              <1>       ; INCREASE PIXEL COLOR
9753                              <1>       ; ecx = pixel count per row
9754                              <1>       ; edi = start pixel address
9755                              <1>
9756                              <1> pix_op_inc_16_0:
9757 0000EEA0 66FF07              <1>       inc   word [edi]
9758 0000EEA3 7503                <1>       jnz   short pix_op_inc_16_1
9759                              <1>       ;mov  word [edi], 0FFFFh ; Max. value
9760 0000EEA5 66FF0F              <1>       dec   word [edi]
9761                              <1> pix_op_inc_16_1:
9762 0000EEA8 47                  <1>       inc   edi
9763 0000EEA9 47                  <1>       inc   edi
9764 0000EEAA E2F4                <1>       loop  pix_op_inc_16_0
9765 0000EEAC C3                  <1>       retn
9766                              <1>
9767                              <1> pix_op_inc_24:
9768                              <1>       ; 24 bit true colors
```

```
9769                                    <1>       ; INCREASE PIXEL COLOR
9770                                    <1>       ; ecx = pixel count per row
9771                                    <1>       ; edi = start pixel address
9772                                    <1>
9773                                    <1> pix_op_inc_24_0:
9774 0000EEAD 8B07                     <1>       mov   eax, [edi]
9775 0000EEAF 40                       <1>       inc   eax
9776 0000EEB0 25FFFFFF00               <1>       and   eax, 0FFFFFFh
9777 0000EEB5 7501                     <1>       jnz   short pix_op_inc_24_1
9778                                    <1>       ;mov  eax, 0FFFFFFh  ; Max. value
9779 0000EEB7 48                       <1>       dec   eax ; 0FFFFFFFFh
9780                                    <1> pix_op_inc_24_1:
9781 0000EEB8 66AB                     <1>       stosw
9782 0000EEBA C1E810                   <1>       shr   eax, 16
9783 0000EEBD AA                       <1>       stosb
9784 0000EEBE E2ED                     <1>       loop  pix_op_inc_24_0
9785 0000EEC0 C3                       <1>       retn
9786                                    <1>
9787                                    <1> pix_op_inc_32:
9788                                    <1>       ; 32 bit true colors
9789                                    <1>       ; INCREASE PIXEL COLOR
9790                                    <1>       ; ecx = pixel count per row
9791                                    <1>       ; edi = start pixel address
9792                                    <1>
9793                                    <1> pix_op_inc_32_0:
9794 0000EEC1 FF07                     <1>       inc   dword [edi]
9795 0000EEC3 7502                     <1>       jnz   short pix_op_inc_32_1
9796                                    <1>       ;mov  dword [edi], 0FFFFFFFFh ; Max. value
9797 0000EEC5 FF0F                     <1>       dec   dword [edi]
9798                                    <1> pix_op_inc_32_1:
9799 0000EEC7 83C704                   <1>       add   edi, 4
9800 0000EECA E2F5                     <1>       loop  pix_op_inc_32_0
9801 0000EECC C3                       <1>       retn
9802                                    <1>
9803                                    <1> pix_op_dec_8:
9804                                    <1>       ; 8 bit colors (256 colors)
9805                                    <1>       ; DECREASE PIXEL COLOR
9806                                    <1>       ; ecx = pixel count per row
9807                                    <1>       ; edi = start pixel address
9808                                    <1>
9809                                    <1> pix_op_dec_8_0:
9810 0000EECD FE0F                     <1>       dec   byte [edi]
9811 0000EECF 7902                     <1>       jns   short pix_op_dec_8_1
9812 0000EED1 FE07                     <1>       inc   byte [edi] ; 0 ; Min. value
9813                                    <1> pix_op_dec_8_1:
9814 0000EED3 47                       <1>       inc   edi
9815 0000EED4 E2F7                     <1>       loop  pix_op_dec_8_0
9816 0000EED6 C3                       <1>       retn
9817                                    <1>
9818                                    <1> pix_op_dec_16:
9819                                    <1>       ; 16 bit colors (65536 colors)
9820                                    <1>       ; DECREASE PIXEL COLOR
9821                                    <1>       ; ecx = pixel count per row
9822                                    <1>       ; edi = start pixel address
9823                                    <1>
9824                                    <1> pix_op_dec_16_0:
9825 0000EED7 66FF0F                   <1>       dec   word [edi]
9826 0000EEDA 7903                     <1>       jns   short pix_op_dec_16_1
9827 0000EEDC 66FF07                   <1>       inc   word [edi] ; 0 ; Min. value
9828                                    <1> pix_op_dec_16_1:
9829 0000EEDF 47                       <1>       inc edi
9830 0000EEE0 47                       <1>       inc edi
9831 0000EEE1 E2F4                     <1>       loop  pix_op_dec_16_0
9832 0000EEE3 C3                       <1>       retn
9833                                    <1>
9834                                    <1> pix_op_dec_24:
9835                                    <1>       ; 24 bit true colors
9836                                    <1>       ; DECREASE PIXEL COLOR
9837                                    <1>       ; ecx = pixel count per row
9838                                    <1>       ; edi = start pixel address
9839                                    <1>
9840                                    <1> pix_op_dec_24_0:
9841 0000EEE4 8B07                     <1>       mov   eax, [edi]
9842 0000EEE6 25FFFFFF00               <1>       and   eax, 0FFFFFFh
9843 0000EEEB 7401                     <1>       jz    short pix_op_dec_24_1
9844                                    <1>                   ; 0 ; Min. value
9845 0000EEED 48                       <1>       dec   eax
9846                                    <1> pix_op_dec_24_1:
9847 0000EEEE 66AB                     <1>       stosw
9848 0000EEF0 C1E810                   <1>       shr   eax, 16
9849 0000EEF3 AA                       <1>       stosb
9850 0000EEF4 E2B7                     <1>       loop  pix_op_inc_24_0
9851 0000EEF6 C3                       <1>       retn
9852                                    <1>
9853                                    <1> pix_op_dec_32:
9854                                    <1>       ; 32 bit true colors
9855                                    <1>       ; DECREASE PIXEL COLOR
9856                                    <1>       ; ecx = pixel count per row
9857                                    <1>       ; edi = start pixel address
9858                                    <1>
9859                                    <1> pix_op_dec_32_0:
9860 0000EEF7 FF0F                     <1>       dec   dword [edi]
9861 0000EEF9 7902                     <1>       jns   short pix_op_dec_32_1
9862 0000EEFB FF07                     <1>       inc   dword [edi] ; 0 ; Min. value
9863                                    <1> pix_op_dec_32_1:
9864 0000EEFD 83C704                   <1>       add   edi, 4
9865 0000EF00 E2F5                     <1>       loop  pix_op_dec_32_0
9866 0000EF02 89D0                     <1>       mov   eax, edx
9867 0000EF04 C3                       <1>       retn
9868                                    <1>
9869                                    <1> pix_op_rpl_8:
9870                                    <1>       ; 8 bit colors (256 colors)
9871                                    <1>       ; REPLACE PIXEL COLORS
9872                                    <1>       ; ecx = pixel count per row
9873                                    <1>       ;  al = new color
```

```
9874                                    <1>       ; byte [maskcolor] = old color
9875                                    <1>       ; edi = start pixel address
9876                                    <1>
9877 0000EF05 8A25[82120300]            <1>       mov    ah, [maskcolor]
9878                                    <1> pix_op_rpl_8_0:
9879 0000EF0B 3A27                      <1>       cmp    ah, [edi]
9880 0000EF0D 7502                      <1>       jne    short pix_op_rpl_8_1
9881 0000EF0F 8807                      <1>       mov    [edi], al
9882                                    <1> pix_op_rpl_8_1:
9883 0000EF11 47                        <1>       inc    edi
9884 0000EF12 E2F7                      <1>       loop   pix_op_rpl_8_0
9885 0000EF14 C3                        <1>       retn
9886                                    <1>
9887                                    <1> pix_op_rpl_16:
9888                                    <1>       ; 16 bit colors (65536 colors)
9889                                    <1>       ; REPLACE PIXEL COLORS
9890                                    <1>       ; ecx = pixel count per row
9891                                    <1>       ;  ax = new color
9892                                    <1>       ; word [maskcolor] = old color
9893                                    <1>       ; edi = start pixel address
9894                                    <1>
9895 0000EF15 8B15[82120300]            <1>       mov    edx, [maskcolor]
9896                                    <1> pix_op_rpl_16_0:
9897 0000EF1B 663B17                    <1>       cmp    dx, [edi]
9898 0000EF1E 7503                      <1>       jne    short pix_op_rpl_16_1
9899 0000EF20 668907                    <1>       mov    [edi], ax
9900                                    <1> pix_op_rpl_16_1:
9901 0000EF23 47                        <1>       inc    edi
9902 0000EF24 47                        <1>       inc    edi
9903 0000EF25 E2F4                      <1>       loop   pix_op_rpl_16_0
9904 0000EF27 C3                        <1>       retn
9905                                    <1>
9906                                    <1> pix_op_rpl_24:
9907                                    <1>       ; 24 bit true colors
9908                                    <1>       ; REPLACE PIXEL COLORS
9909                                    <1>       ; ecx = pixel count per row
9910                                    <1>       ; eax = new color
9911                                    <1>       ; [maskcolor] = old color
9912                                    <1>       ; edi = start pixel address
9913                                    <1>
9914                                    <1> pix_op_rpl_24_0:
9915 0000EF28 8B17                      <1>       mov    edx, [edi]
9916 0000EF2A 81E2FFFFFF00              <1>       and    edx, 0FFFFFFh
9917 0000EF30 3B15[82120300]            <1>       cmp    edx, [maskcolor]
9918 0000EF36 7406                      <1>       je     short pix_op_rpl_24_1
9919 0000EF38 83C703                    <1>       add    edi, 3
9920 0000EF3B E2EB                      <1>       loop   pix_op_rpl_24_0
9921 0000EF3D C3                        <1>       retn
9922                                    <1> pix_op_rpl_24_1:
9923 0000EF3E AA                        <1>       stosb
9924 0000EF3F C1C808                    <1>       ror    eax, 8
9925 0000EF42 66AB                      <1>       stosw
9926 0000EF44 C1C008                    <1>       rol    eax, 8
9927 0000EF47 E2DF                      <1>       loop   pix_op_rpl_24_0
9928 0000EF49 C3                        <1>       retn
9929                                    <1>
9930                                    <1> pix_op_rpl_32:
9931                                    <1>       ; 32 bit true colors
9932                                    <1>       ; REPLACE PIXEL COLORS
9933                                    <1>       ; ecx = pixel count per row
9934                                    <1>       ; eax = new color
9935                                    <1>       ; [maskcolor] = old color
9936                                    <1>       ; edi = start pixel address
9937                                    <1>
9938 0000EF4A 8B15[82120300]            <1>       mov    edx, [maskcolor]
9939                                    <1> pix_op_rpl_32_0:
9940 0000EF50 3B17                      <1>       cmp    edx, [edi]
9941 0000EF52 7504                      <1>       jne    short pix_op_rpl_32_2
9942 0000EF54 AB                        <1>       stosd
9943 0000EF55 E2F9                      <1>       loop   pix_op_rpl_32_0
9944 0000EF57 C3                        <1>       retn
9945                                    <1> pix_op_rpl_32_2:
9946 0000EF58 83C704                    <1>       add    edi, 4
9947 0000EF5B E2F3                      <1>       loop   pix_op_rpl_32_0
9948 0000EF5D C3                        <1>       retn
9949                                    <1>
9950                                    <1> pix_op_chr:
9951                                    <1>       ; 15/02/2021
9952                                    <1>       ; 05/02/2021
9953                                    <1>       ; WRITE CHARACTER (FONT)
9954                                    <1>       ; 05/01/2021 ([ufont])
9955                                    <1>       ; 01/01/2021
9956                                    <1>       ;     CL = char's color (8 bit, 256 colors)
9957                                    <1>       ;    ECX = char's color (16 bit and true colors)
9958                                    <1>       ;     DL = Character's ASCII code
9959                                    <1>       ;     DH bit 0 -> font height
9960                                    <1>       ;        0 -> 8x16 character font
9961                                    <1>       ;        1 -> 8x8 character font
9962                                    <1>       ;     DH bit 1 & 2 -> scale
9963                                    <1>       ;        0 = 1/1 (8 pixels per char row)
9964                                    <1>       ;        1 = 2/1 (16 pixels per char row)
9965                                    <1>       ;        2 = 3/1 (24 pixels per char row)
9966                                    <1>       ;        3 = 4/1 (32 pixels per char row)
9967                                    <1>       ;     DH bit 6 -> [ufont] option (1 = use [ufont])
9968                                    <1>       ;     If DH bit 7 = 1
9969                                    <1>       ;       USER FONT (from user buffer)
9970                                    <1>       ;          DL = 0 -> 8x8 (width: 1 byte per row)
9971                                    <1>       ;          DL = 1 -> 8x16
9972                                    <1>       ;          DL = 2 -> 16x16 (width: 2 bytes)
9973                                    <1>       ;          DL = 3 -> 16x32
9974                                    <1>       ;          DL = 4 -> 24x24 (width: 3 bytes)
9975                                    <1>       ;          DL = 5 -> 24x48
9976                                    <1>       ;          DL = 6 -> 32x32 (width: 4 bytes)
9977                                    <1>       ;          DL = 7 -> 32x64
9978                                    <1>       ;          DL > 7 -> invalid (unused)
```

```
 9979                           <1>      ;       EDI = user's font buffer address
 9980                           <1>      ;          (NOTE: byte order is as row0,row1,row2..)
 9981                           <1>      ;     ESI = start position (row, column) (*)
 9982                           <1>      ;           (HW = row, SI = column)
 9983                           <1>
 9984 0000EF5E 89F0             <1>      mov    eax, esi ; start position
 9985 0000EF60 E855F2FFFF       <1>      call   calc_pixel_offset
 9986 0000EF65 3B05[76120300]   <1>      cmp    eax, [v_siz]
 9987 0000EF6B 736D             <1>      jnb    short pix_op_chr_err ; out of display page!
 9988 0000EF6D E825F2FFFF       <1>      call   pixels_to_byte_count
 9989                           <1>      ; eax = font start offset
 9990 0000EF72 0305[72120300]   <1>      add    eax, [v_mem] ; LFB start address
 9991 0000EF78 A3[7A120300]     <1>      mov    [v_str], eax ; font start address
 9992                           <1>
 9993 0000EF7D 890D[82120300]   <1>      mov    [maskcolor], ecx ; save char's color
 9994                           <1>
 9995 0000EF83 8835[70120300]   <1>      mov    [v_ops], dh
 9996                           <1>
 9997 0000EF89 81E6FFFF0000     <1>      and    esi, 0FFFFh
 9998 0000EF8F 8935[8A120300]   <1>      mov    [buffer8], esi ; start column
 9999                           <1>
10000 0000EF95 31DB             <1>      xor    ebx, ebx ; 0
10001 0000EF97 31C0             <1>      xor    eax, eax ; 15/02/2021
10002                           <1>
10003 0000EF99 F6C680           <1>      test   dh, 80h
10004 0000EF9C 7577             <1>      jnz    short pix_op_chr_u ; user font
10005                           <1>
10006 0000EF9E 80E63F           <1>      and    dh, 3Fh ; clear bit 6, [UFONT] option bit
10007 0000EFA1 7409             <1>      jz     short pix_op_chr_0
10008                           <1>
10009 0000EFA3 80FE07           <1>      cmp    dh, 7
10010 0000EFA6 7732             <1>      ja     short pix_op_chr_err
10011                           <1>                    ; invalid (undefined) option
10012 0000EFA8 88F4             <1>      mov    ah, dh
10013 0000EFAA D0EC             <1>      shr    ah, 1
10014                           <1>      ; ah = 0 to 3, scale
10015                           <1>      ;jmp   short pix_op_chr_font_pixels
10016                           <1>
10017                           <1> pix_op_chr_font_pixels:
10018                           <1>      ; 05/02/2021
10019                           <1>      ; write scaled font to buffer
10020                           <1>
10021                           <1>      ; DL = ASCII code of character
10022                           <1>      ; AH = scale
10023                           <1>      ; EDI = buffer address (kernel)
10024                           <1>
10025                           <1> pix_op_chr_0:
10026 0000EFAC 88D3             <1>      mov    bl, dl ; 15/02/2021
10027 0000EFAE 31C9             <1>      xor    ecx, ecx
10028 0000EFB0 B610             <1>      mov    dh, 16
10029 0000EFB2 F605[70120300]01 <1>      test   byte [v_ops], 1 ; 8x8 font ?
10030 0000EFB9 7428             <1>      jz     short pix_op_chr_2 ; 8x16 font
10031 0000EFBB B608             <1>      mov    dh, 8
10032 0000EFBD C1E303           <1>      shl    ebx, 3 ; * 8
10033 0000EFC0 F605[70120300]40 <1>      test   byte [v_ops], 40h ; [ufont] option
10034 0000EFC7 7412             <1>      jz     short pix_op_chr_1  ; no
10035                           <1>      ; test 8x8 user font is ready flag
10036 0000EFC9 F605[66120300]01 <1>      test   byte [ufont], 1
10037 0000EFD0 7409             <1>      jz     short pix_op_chr_1 ; no
10038 0000EFD2 81C300500900     <1>      add    ebx, VGAFONT8USER
10039 0000EFD8 EB2C             <1>      jmp    short pix_op_chr_fpos_0
10040                           <1> pix_op_chr_err:
10041 0000EFDA C3               <1>      retn
10042                           <1> pix_op_chr_1:
10043 0000EFDB 81C3[3C570100]   <1>      add    ebx, vgafont8 ; system font (8x8)
10044 0000EFE1 EB23             <1>      jmp    short pix_op_chr_fpos_0
10045                           <1> pix_op_chr_2:
10046 0000EFE3 C1E304           <1>      shl    ebx, 4 ; * 16
10047 0000EFE6 F605[70120300]40 <1>      test   byte [v_ops], 40h ; [ufont] option
10048 0000EFED 7411             <1>      jz     short pix_op_chr_3  ; no
10049                           <1>      ; test 8x16 user font is ready flag
10050 0000EFEF F605[66120300]02 <1>      test   byte [ufont], 2
10051 0000EFF6 7408             <1>      jz     short pix_op_chr_3 ; no
10052 0000EFF8 81C300400900     <1>      add    ebx, VGAFONT16USER
10053 0000EFFE EB06             <1>      jmp    short pix_op_chr_fpos_0
10054                           <1> pix_op_chr_3:
10055 0000F000 81C3[3C6D0100]   <1>      add    ebx, vgafont16 ; system font (8x16)
10056                           <1> pix_op_chr_fpos_0:
10057 0000F006 20E4             <1>      and    ah, ah
10058 0000F008 754B             <1>      jnz    short pix_op_chr_fpos_1 ; scale > 1
10059                           <1>      ; no scale (scale = 1)
10060 0000F00A 89DE             <1>      mov    esi, ebx ; 15/02/2021
10061 0000F00C 88F1             <1>      mov    cl, dh ; rows/height (16 or 8)
10062 0000F00E B608             <1>      mov    dh, 8  ; columns/width
10063 0000F010 E9CD000000       <1>      jmp    pix_op_chr_f2p
10064                           <1> pix_op_chr_u:
10065                           <1>      ; write user defined font
10066 0000F015 80FE80           <1>      cmp    dh, 80h
10067 0000F018 75C0             <1>      jne    short pix_op_chr_err
10068 0000F01A 80FA07           <1>      cmp    dl, 7
10069 0000F01D 77BB             <1>      ja     short pix_op_chr_err
10070                           <1>
10071                           <1>      ; 16/02/2021
10072 0000F01F 89FE             <1>      mov    esi, edi ; user's font buffer
10073                           <1>
10074                           <1>      ;xor   eax, eax
10075                           <1>      ; eax = 0 ; 15/02/2021
10076 0000F021 88D4             <1>      mov    ah, dl
10077 0000F023 D0EC             <1>      shr    ah, 1
10078 0000F025 FEC4             <1>      inc    ah
10079                           <1>      ; ah =  1 to 4
10080 0000F027 88E0             <1>      mov    al, ah
10081 0000F029 C0E003           <1>      shl    al, 3 ; * 8
10082                           <1>      ; al = 8,16,24,32
10083 0000F02C 88C3             <1>      mov    bl, al
```

```
10084 0000F02E 88C7              <1>       mov    bh, al
10085 0000F030 F6E4              <1>       mul    ah
10086                            <1>       ; ax = 8,32,72,128 bytes
10087 0000F032 F6C201            <1>       test   dl, 1
10088 0000F035 7405              <1>       jz     short pix_op_chr_u_0
10089 0000F037 66D1E0            <1>       shl    ax, 1 ; *2
10090                            <1>       ; ax = 16,32,144,256 bytes
10091 0000F03A D0E7              <1>       shl    bh, 1
10092                            <1> pix_op_chr_u_0:
10093                            <1>       ; eax = byte count
10094 0000F03C 89C1              <1>       mov    ecx, eax
10095 0000F03E BF00760900        <1>       mov    edi, VBE3SAVERESTOREBLOCK
10096                            <1>       ; esi = user buffer
10097 0000F043 E844250000        <1>       call   transfer_from_user_buffer
10098 0000F048 7290              <1>       jc     short pix_op_chr_err
10099                            <1>
10100 0000F04A 88F9              <1>       mov    cl, bh ; rows/height
10101 0000F04C 88DE              <1>       mov    dh, bl ; columns (width)
10102 0000F04E 89FE              <1>       mov    esi, edi ; VBE3SAVERESTOREBLOCK
10103 0000F050 E98D000000        <1>       jmp    pix_op_chr_f2p
10104                            <1>
10105                            <1> pix_op_chr_fpos_1:
10106                            <1>       ; 18/02/2021
10107                            <1>       ; scale > 1
10108 0000F055 88F5              <1>       mov    ch, dh ; 16 or 8
10109 0000F057 BF00760900        <1>       mov    edi, VBE3SAVERESTOREBLOCK
10110 0000F05C 89FE              <1>       mov    esi, edi
10111 0000F05E FECC              <1>       dec    ah
10112 0000F060 7523              <1>       jnz    short pix_op_chr_fpos_5 ; scale > 2
10113                            <1>       ; scale = 2
10114                            <1> pix_op_chr_fpos_2:
10115 0000F062 B108              <1>       mov    cl, 8
10116 0000F064 8A13              <1>       mov    dl, [ebx]
10117                            <1> pix_op_chr_fpos_3:
10118 0000F066 66C1E002          <1>       shl    ax, 2
10119 0000F06A D0E2              <1>       shl    dl, 1
10120 0000F06C 7302              <1>       jnc    short pix_op_chr_fpos_4
10121 0000F06E 0C03              <1>       or     al, 3
10122                            <1> pix_op_chr_fpos_4:
10123 0000F070 FEC9              <1>       dec    cl
10124 0000F072 75F2              <1>       jnz    short pix_op_chr_fpos_3
10125 0000F074 66AB              <1>       stosw
10126                            <1>       ; 18/02/2021
10127 0000F076 66AB              <1>       stosw
10128 0000F078 43                <1>       inc    ebx
10129 0000F079 FECD              <1>       dec    ch
10130 0000F07B 75E5              <1>       jnz    short pix_op_chr_fpos_2
10131                            <1>       ; scale = 2
10132 0000F07D 88F1              <1>       mov    cl, dh ; 16 or 8 (height/rows)
10133 0000F07F D0E1              <1>       shl    cl, 1  ; 32 or 16 rows
10134 0000F081 B610              <1>       mov    dh, 16 ; columns (width)
10135 0000F083 EB5D              <1>       jmp    short pix_op_chr_f2p
10136                            <1> pix_op_chr_fpos_5:
10137 0000F085 FECC              <1>       dec    ah
10138 0000F087 7538              <1>       jnz    short pix_op_chr_fpos_9 ; scale = 4
10139                            <1>       ; scale = 3
10140                            <1> pix_op_chr_fpos_6:
10141 0000F089 B108              <1>       mov    cl, 8
10142 0000F08B 8A13              <1>       mov    dl, [ebx]
10143                            <1> pix_op_chr_fpos_7:
10144 0000F08D C1E003            <1>       shl    eax, 3
10145 0000F090 D0E2              <1>       shl    dl, 1 ; 18/02/2021
10146 0000F092 7302              <1>       jnc    short pix_op_chr_fpos_8
10147 0000F094 0C07              <1>       or     al, 7
10148                            <1> pix_op_chr_fpos_8:
10149 0000F096 FEC9              <1>       dec    cl
10150 0000F098 75F3              <1>       jnz    short pix_op_chr_fpos_7
10151 0000F09A 66AB              <1>       stosw
10152                            <1>       ; 18/02/2021
10153 0000F09C C1C810            <1>       ror    eax, 16
10154 0000F09F AA                <1>       stosb
10155 0000F0A0 C1C010            <1>       rol    eax, 16
10156 0000F0A3 66AB              <1>       stosw
10157 0000F0A5 C1C810            <1>       ror    eax, 16
10158 0000F0A8 AA                <1>       stosb
10159 0000F0A9 C1C010            <1>       rol    eax, 16
10160 0000F0AC 66AB              <1>       stosw
10161 0000F0AE C1E810            <1>       shr    eax, 16 ; 27/02/2021
10162 0000F0B1 AA                <1>       stosb
10163 0000F0B2 43                <1>       inc    ebx
10164                            <1>       ; 18/02/2021
10165 0000F0B3 FECD              <1>       dec    ch
10166 0000F0B5 75D2              <1>       jnz    short pix_op_chr_fpos_6
10167                            <1>       ; scale = 3
10168 0000F0B7 88F1              <1>       mov    cl, dh ; 16 or 8 (height/rows)
10169 0000F0B9 D0E1              <1>       shl    cl, 1
10170 0000F0BB 00F1              <1>       add    cl, dh ; 48 or 24 rows
10171 0000F0BD B618              <1>       mov    dh, 24 ; columns (width)
10172 0000F0BF EB21              <1>       jmp    short pix_op_chr_f2p
10173                            <1>
10174                            <1> pix_op_chr_fpos_9:
10175                            <1>       ; scale = 4
10176 0000F0C1 B108              <1>       mov    cl, 8
10177 0000F0C3 8A13              <1>       mov    dl, [ebx]
10178                            <1> pix_op_chr_fpos_10:
10179                            <1>       ; 18/02/2021
10180 0000F0C5 C1E004            <1>       shl    eax, 4
10181 0000F0C8 D0E2              <1>       shl    dl, 1 ; 18/02/2021
10182 0000F0CA 7302              <1>       jnc    short pix_op_chr_fpos_11
10183 0000F0CC 0C0F              <1>       or     al, 0Fh ; or al, 15
10184                            <1> pix_op_chr_fpos_11:
10185 0000F0CE FEC9              <1>       dec    cl
10186 0000F0D0 75F3              <1>       jnz    short pix_op_chr_fpos_10
10187 0000F0D2 AB                <1>       stosd
10188                            <1>       ; 18/02/2021
```

```
10189 0000F0D3 AB                  <1>       stosd
10190 0000F0D4 AB                  <1>       stosd
10191 0000F0D5 AB                  <1>       stosd
10192 0000F0D6 43                  <1>       inc    ebx
10193 0000F0D7 FECD                <1>       dec    ch
10194 0000F0D9 75E6                <1>       jnz    short pix_op_chr_fpos_9
10195                              <1>       ; scale = 4
10196 0000F0DB 88F1                <1>       mov    cl, dh ; 16 or 8 (height/rows)
10197 0000F0DD C0E102              <1>       shl    cl, 2 ; 64 or 32 rows
10198 0000F0E0 B620                <1>       mov    dh, 32 ; columns (width)
10199                              <1>       ;jmp   short pix_op_chr_f2p
10200                              <1>
10201                              <1> pix_op_chr_f2p:
10202                              <1>       ; write font pixels
10203 0000F0E2 8B3D[7A120300]      <1>       mov    edi, [v_str]
10204                              <1>       ; 15/02/2021
10205                              <1> pix_op_chr_f2p_next:
10206 0000F0E8 80FE08              <1>       cmp    dh, 8
10207 0000F0EB 7706                <1>       ja     short pix_op_chr_f2p_24
10208                              <1> pix_op_chr_f2p_8:
10209 0000F0ED AC                  <1>       lodsb
10210 0000F0EE C1E018              <1>       shl    eax, 24 ; 15/02/2021
10211 0000F0F1 EB16                <1>       jmp    short pix_op_chr_f2p_0
10212                              <1> pix_op_chr_f2p_24:
10213 0000F0F3 80FE18              <1>       cmp    dh, 24
10214 0000F0F6 7710                <1>       ja     short pix_op_chr_f2p_32
10215 0000F0F8 7207                <1>       jb     short pix_op_chr_f2p_16
10216                              <1>       ; 27/02/2021
10217                              <1>       ;mov   eax, [esi]
10218 0000F0FA AD                  <1>       lodsd
10219 0000F0FB C1E008              <1>       shl    eax, 8
10220                              <1>       ;add   esi, 3
10221 0000F0FE 4E                  <1>       dec    esi
10222 0000F0FF EB08                <1>       jmp    short pix_op_chr_f2p_0
10223                              <1> pix_op_chr_f2p_16:
10224 0000F101 66AD                <1>       lodsw
10225 0000F103 C1E010              <1>       shl    eax, 16 ; 15/02/2021
10226 0000F106 EB01                <1>       jmp    short pix_op_chr_f2p_0
10227                              <1> pix_op_chr_f2p_32:
10228 0000F108 AD                  <1>       lodsd
10229                              <1> pix_op_chr_f2p_0:
10230                              <1>       ; EAX = font row (8,16,24,32 pixels)
10231                              <1>       ;      (bits are shifted to left)
10232                              <1>       ; CL = rows
10233                              <1>       ; DH = bits per row (8,16,24,32)
10234 0000F109 8B1D[8A120300]      <1>       mov    ebx, [buffer8] ; start column
10235 0000F10F 57                  <1>       push   edi ; *
10236 0000F110 52                  <1>       push   edx ; **
10237                              <1> pix_op_chr_f2p_1:
10238 0000F111 E82E000000          <1>       call   pix_op_chr_w_pixel
10239                              <1> pix_op_chr_f2p_2:
10240 0000F116 663B1D[6E120300]    <1>       cmp    bx, [v_width] ; current column
10241 0000F11D 7304                <1>       jnb    short pix_op_chr_f2p_3
10242 0000F11F FECE                <1>       dec    dh
10243 0000F121 75EE                <1>       jnz    short pix_op_chr_f2p_1 ; next bit
10244                              <1> pix_op_chr_f2p_3:
10245                              <1>       ;mov   ebx, [buffer8]
10246 0000F123 5A                  <1>       pop    edx ; **
10247 0000F124 58                  <1>       pop    eax ; *
10248 0000F125 3B3D[7E120300]      <1>       cmp    edi, [v_end]
10249 0000F12B 7316                <1>       jnb    short pix_op_chr_f2p_4
10250 0000F12D FEC9                <1>       dec    cl
10251 0000F12F 7412                <1>       jz     short pix_op_chr_f2p_4
10252                              <1>       ; 27/02/2021
10253 0000F131 89C7                <1>       mov    edi, eax
10254 0000F133 0FB705[6E120300]    <1>       movzx  eax, word [v_width]
10255 0000F13A E858F0FFFF          <1>       call   pixels_to_byte_count
10256 0000F13F 01C7                <1>       add    edi, eax ; next position
10257 0000F141 EBA5                <1>       jmp    short pix_op_chr_f2p_next
10258                              <1> pix_op_chr_f2p_4:
10259 0000F143 C3                  <1>       retn
10260                              <1>
10261                              <1> pix_op_chr_w_pixel:
10262                              <1>       ; 15/02/2021
10263 0000F144 89C5                <1>       mov    ebp, eax
10264 0000F146 A1[82120300]        <1>       mov    eax, [maskcolor]
10265 0000F14B 803D[71120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
10266 0000F152 7711                <1>       ja     short pix_op_chr_wp_2
10267                              <1>       ; 256 colors (1 byte per pixel)
10268 0000F154 D1E5                <1>       shl    ebp, 1
10269 0000F156 7302                <1>       jnc    short pix_op_chr_wp_0
10270 0000F158 8807                <1>       mov    [edi], al
10271                              <1> pix_op_chr_wp_0:
10272 0000F15A 47                  <1>       inc    edi
10273 0000F15B FF05[64030300]      <1>       inc    dword [u.r0] ; +1
10274                              <1> pix_op_chr_wp_1:
10275 0000F161 43                  <1>       inc    ebx
10276 0000F162 89E8                <1>       mov    eax, ebp
10277 0000F164 C3                  <1>       retn
10278                              <1> pix_op_chr_wp_2:
10279 0000F165 803D[71120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
10280 0000F16C 772E                <1>       ja     short pix_op_chr_wp_6 ; 32bpp
10281 0000F16E 721C                <1>       jb     short pix_op_chr_wp_4 ; 16bpp
10282                              <1>       ; 24 bit true colors
10283                              <1>       ; * 3
10284 0000F170 D1E5                <1>       shl    ebp, 1
10285 0000F172 7309                <1>       jnc    short pix_op_chr_wp_3
10286 0000F174 668907              <1>       mov    [edi], ax
10287 0000F177 C1E810              <1>       shr    eax, 16
10288 0000F17A 884702              <1>       mov    [edi+2], al
10289                              <1> pix_op_chr_wp_3:
10290 0000F17D B803000000          <1>       mov    eax, 3 ; 27/02/2021
10291 0000F182 01C7                <1>       add    edi, eax  ; add edi, 3
10292 0000F184 0105[64030300]      <1>       add    [u.r0], eax ; +3
10293                              <1>
```

```
10294 0000F18A EBD5               <1>       jmp    short pix_op_chr_wp_1
10295                             <1>
10296                             <1> pix_op_chr_wp_4:
10297                             <1>       ; 16 bit (65536) colors
10298 0000F18C D1E5               <1>       shl    ebp, 1
10299 0000F18E 7303               <1>       jnc    short pix_op_chr_wp_5
10300 0000F190 668907             <1>       mov    [edi], ax
10301                             <1> pix_op_chr_wp_5:
10302 0000F193 47                 <1>       inc    edi
10303 0000F194 FF05[64030300]     <1>       inc    dword [u.r0] ; +1
10304 0000F19A EBBE               <1>       jmp    short pix_op_chr_wp_0
10305                             <1>
10306                             <1> pix_op_chr_wp_6:
10307                             <1>       ; 32 bit true colors
10308 0000F19C D1E5               <1>       shl    ebp, 1
10309 0000F19E 7302               <1>       jnc    short pix_op_chr_wp_7
10310 0000F1A0 8907               <1>       mov    [edi], eax
10311                             <1> pix_op_chr_wp_7:
10312 0000F1A2 31C0               <1>       xor    eax, eax
10313 0000F1A4 B004               <1>       mov    al, 4
10314 0000F1A6 01C7               <1>       add    edi, eax ; add edi, 4
10315 0000F1A8 0105[64030300]     <1>       add    [u.r0], eax ; +4
10316 0000F1AE EBB1               <1>       jmp    short pix_op_chr_wp_1
10317                             <1>
10318                             <1> m_pix_op_cpy:
10319                             <1>       ; 26/02/2021
10320                             <1>       ; 06/02/2021
10321                             <1>       ; MASKED COPY PIXELS (full screen)
10322                             <1>       ;
10323                             <1>       ; jump from pix_op_cpy
10324                             <1>       ;
10325                             <1>       ; INPUT:
10326                             <1>       ;   ecx = transfer count (bytes)
10327                             <1>       ;   edi = [v_mem] = start address of LFB
10328                             <1>       ;   esi = user's buffer address (virtual)
10329                             <1>       ;
10330                             <1>       ; OUTPUT:
10331                             <1>       ;   [u.r0] will be > 0 if succesful
10332                             <1>
10333                             <1>       ; Full screen masked copy
10334                             <1>
10335                             <1>       ; Modified regs: eax, ebx, edx, esi, edi, ecx
10336                             <1>
10337                             <1> m_pix_op_cpy_0:
10338                             <1>       ;push  ebx ; *** ; 26/02/2021
10339 0000F1B0 57                 <1>       push   edi ; **
10340 0000F1B1 51                 <1>       push   ecx ; *
10341 0000F1B2 81F9F8070000       <1>       cmp    ecx, 2040 ; (3*680) ; 26/02/2021
10342 0000F1B8 7605               <1>       jna    short m_pix_op_cpy_1
10343 0000F1BA B9F8070000         <1>       mov    ecx, 2040
10344                             <1> m_pix_op_cpy_1:
10345 0000F1BF BF00760900         <1>       mov    edi, VBE3SAVERESTOREBLOCK ; temporary buff
10346 0000F1C4 E8C3230000         <1>       call   transfer_from_user_buffer
10347 0000F1C9 726C               <1>       jc     short m_pix_op_cpy_3
10348 0000F1CB 01CE               <1>       add    esi, ecx
10349 0000F1CD 89F5               <1>       mov    ebp, esi  ; save user's buffer address
10350 0000F1CF 89FE               <1>       mov    esi, edi
10351 0000F1D1 89CB               <1>       mov    ebx, ecx
10352 0000F1D3 59                 <1>       pop    ecx ; *
10353 0000F1D4 29D9               <1>       sub    ecx, ebx
10354 0000F1D6 5F                 <1>       pop    edi ; **
10355 0000F1D7 31D2               <1>       xor    edx, edx ; 26/02/2021
10356 0000F1D9 803D[71120300]08   <1>       cmp    byte [v_bpp], 8
10357 0000F1E0 7435               <1>       je     short m_pix_op_cpy_1_8
10358 0000F1E2 803D[71120300]18   <1>       cmp    byte [v_bpp], 24
10359 0000F1E9 776D               <1>       ja     short m_pix_op_cpy_1_32
10360 0000F1EB 724D               <1>       jb     short m_pix_op_cpy_1_16
10361                             <1> m_pix_op_cpy_1_24:
10362                             <1>       ; 24 bit masked copy
10363                             <1>       ;mov   edx, 3
10364 0000F1ED B203               <1>       mov    dl, 3 ; 26/02/2021
10365                             <1> m_pix_op_cpy_1_24_0:
10366 0000F1EF 66AD               <1>       lodsw
10367 0000F1F1 C1E010             <1>       shl    eax, 16
10368 0000F1F4 AC                 <1>       lodsb
10369 0000F1F5 C1C010             <1>       rol    eax, 16
10370 0000F1F8 3B05[82120300]     <1>       cmp    eax, [maskcolor]
10371 0000F1FE 740F               <1>       je     short m_pix_op_cpy_1_24_1 ; exclude
10372 0000F200 668907             <1>       mov    [edi], ax
10373 0000F203 C1E810             <1>       shr    eax, 16
10374 0000F206 884702             <1>       mov    [edi+2], al
10375 0000F209 0115[64030300]     <1>       add    [u.r0], edx ; +3
10376                             <1> m_pix_op_cpy_1_24_1:
10377 0000F20F 01D7               <1>       add    edi, edx ; +3
10378 0000F211 29D3               <1>       sub    ebx, edx ; sub ebx, 3
10379 0000F213 77DA               <1>       ja     short m_pix_op_cpy_1_24_0
10380 0000F215 EB15               <1>       jmp    short m_pix_op_cpy_2
10381                             <1>
10382                             <1> m_pix_op_cpy_1_8:
10383                             <1>       ; 8 bit masked copy
10384 0000F217 AC                 <1>       lodsb
10385 0000F218 3A05[82120300]     <1>       cmp    al, [maskcolor]
10386 0000F21E 7408               <1>       je     short m_pix_op_cpy_1_8_1 ; exclude
10387 0000F220 8807               <1>       mov    [edi], al
10388 0000F222 FF05[64030300]     <1>       inc    dword [u.r0] ; +1
10389                             <1> m_pix_op_cpy_1_8_1:
10390 0000F228 47                 <1>       inc    edi ; +1
10391 0000F229 4B                 <1>       dec    ebx
10392 0000F22A 75EB               <1>       jnz    short m_pix_op_cpy_1_8
10393                             <1> m_pix_op_cpy_2:
10394 0000F22C 89EE               <1>       mov    esi, ebp ; restore user's buffer addr
10395 0000F22E 09C9               <1>       or     ecx, ecx
10396                             <1>       ; 26/02/2021
10397 0000F230 7407               <1>       jz     short m_pix_of_cpy_4
10398 0000F232 E979FFFFFF         <1>       jmp    m_pix_op_cpy_0
```

```
10399                                  <1> m_pix_op_cpy_3:
10400 0000F237 59                      <1>         pop    ecx ; *
10401 0000F238 5F                      <1>         pop    edi ; **
10402                                  <1> m_pix_of_cpy_4:
10403                                  <1>         ;pop   ebx ; *** ; 26/02/2021
10404 0000F239 C3                      <1>         retn
10405                                  <1>
10406                                  <1> m_pix_op_cpy_1_16:
10407                                  <1>         ; 16 bit masked copy
10408                                  <1>         ;mov   edx, 2
10409 0000F23A B202                    <1>         mov    dl, 2 ; 26/02/2021
10410                                  <1> m_pix_op_cpy_1_16_0:
10411 0000F23C 66AD                    <1>         lodsw
10412 0000F23E 663B05[82120300]        <1>         cmp    ax, [maskcolor]
10413 0000F245 7409                    <1>         je     short m_pix_op_cpy_1_16_1 ; exclude
10414 0000F247 668907                  <1>         mov    [edi], ax
10415 0000F24A 0115[64030300]          <1>         add    [u.r0], edx ; +2
10416                                  <1> m_pix_op_cpy_1_16_1:
10417 0000F250 01D7                    <1>         add    edi, edx ; +2
10418 0000F252 29D3                    <1>         sub    ebx, edx ; sub ebx, 2
10419 0000F254 77E6                    <1>         ja     short m_pix_op_cpy_1_16_0
10420 0000F256 EBD4                    <1>         jmp    short m_pix_op_cpy_2
10421                                  <1>
10422                                  <1> m_pix_op_cpy_1_32:
10423                                  <1>         ; 32 bit masked copy
10424                                  <1>         ;mov   edx, 4
10425 0000F258 B204                    <1>         mov    dl, 4 ; 26/02/2021
10426                                  <1> m_pix_op_cpy_1_32_0:
10427 0000F25A AD                      <1>         lodsd
10428 0000F25B 3B05[82120300]          <1>         cmp    eax, [maskcolor]
10429 0000F261 7408                    <1>         je     short m_pix_op_cpy_1_32_1 ; exclude
10430 0000F263 8907                    <1>         mov    [edi], eax
10431 0000F265 0115[64030300]          <1>         add    [u.r0], edx ; +4
10432                                  <1> m_pix_op_cpy_1_32_1:
10433 0000F26B 01D7                    <1>         add    edi, edx ; +4
10434 0000F26D 29D3                    <1>         sub    ebx, edx ; sub ebx, 4
10435 0000F26F 77E9                    <1>         ja     short m_pix_op_cpy_1_32_0
10436 0000F271 EBB9                    <1>         jmp    short m_pix_op_cpy_2
10437                                  <1>
10438                                  <1> m_pix_op_cpy_w:
10439                                  <1>         ; 26/02/2021
10440                                  <1>         ; 06/02/2021
10441                                  <1>         ; MASKED COPY PIXELS (window)
10442                                  <1>         ;
10443                                  <1>         ; jump from pix_op_cpy_w
10444                                  <1>         ;
10445                                  <1>         ; INPUT:
10446                                  <1>         ;   ecx = bytes per row (to be applied)
10447                                  <1>         ;   edx = screen width in bytes
10448                                  <1>         ;   ebx = row count
10449                                  <1>         ;   esi = user's buffer address
10450                                  <1>         ;   [v_str] = window start address
10451                                  <1>         ;
10452                                  <1>         ; OUTPUT:
10453                                  <1>         ;   [u.r0] will be > 0 if succesful
10454                                  <1>
10455                                  <1>         ; Window masked copy
10456                                  <1>
10457                                  <1> m_pix_op_cpy_w_0:
10458 0000F273 8B3D[7A120300]          <1>         mov    edi, [v_str]
10459                                  <1> m_pix_op_cpy_w_1:
10460 0000F279 57                      <1>         push   edi
10461 0000F27A 56                      <1>         push   esi
10462 0000F27B 53                      <1>         push   ebx
10463 0000F27C 52                      <1>         push   edx
10464 0000F27D 51                      <1>         push   ecx
10465 0000F27E E82DFFFFFF              <1>         call   m_pix_op_cpy ; 26/02/2021
10466 0000F283 59                      <1>         pop    ecx
10467 0000F284 5A                      <1>         pop    edx
10468 0000F285 5B                      <1>         pop    ebx
10469 0000F286 5E                      <1>         pop    esi
10470 0000F287 5F                      <1>         pop    edi
10471 0000F288 7209                    <1>         jc     short m_pix_op_cpy_w_2
10472 0000F28A 4B                      <1>         dec    ebx
10473 0000F28B 7406                    <1>         jz     short m_pix_op_cpy_w_2 ; ok.
10474                                  <1>         ; next row
10475 0000F28D 01CE                    <1>         add    esi, ecx ; next row in user's buffer
10476 0000F28F 01D7                    <1>         add    edi, edx ; next row of window (system)
10477 0000F291 EBE6                    <1>         jmp    short m_pix_op_cpy_w_1
10478                                  <1> m_pix_op_cpy_w_2:
10479 0000F293 C3                      <1>         retn
10480                                  <1>
10481                                  <1> m_pix_op_new:
10482                                  <1>         ; 06/02/2021
10483                                  <1>         ; CHANGE COLOR (MASKED, full screen)
10484                                  <1>         ;
10485                                  <1>         ; jump from pix_op_new
10486                                  <1>         ;
10487                                  <1>         ; INPUT:
10488                                  <1>         ;   eax = color (AL, AX, EAX)
10489                                  <1>         ;   ecx = [v_siz] ; display page pixel count
10490                                  <1>         ;   esi = edi = [v_mem] ; LFB start address
10491                                  <1>         ;
10492                                  <1>         ;   [maskcolor] = mask color (to be excluded)
10493                                  <1>         ;
10494                                  <1>         ; OUTPUT:
10495                                  <1>         ;      [u.r0] will be > 0 if succesful
10496                                  <1>
10497                                  <1>         ; Full screen
10498                                  <1> m_pix_op_new_0:
10499 0000F294 803D[71120300]08        <1>         cmp    byte [v_bpp], 8 ; 8bpp
10500 0000F29B 7717                    <1>         ja     short m_pix_op_new_1
10501                                  <1>         ; 256 colors (8bpp)
10502                                  <1>         ;jmp   short m_pix_op_new_8
10503                                  <1> m_pix_op_new_8:
```

```
10504                                  <1>       ; 8 bit colors (256 colors)
10505 0000F29D 88C2                    <1>       mov    dl, al ; new color
10506                                  <1> m_pix_op_new_8_0:
10507 0000F29F AC                      <1>       lodsb
10508 0000F2A0 3A05[82120300]          <1>       cmp    al, [maskcolor]
10509 0000F2A6 7408                    <1>       je     short m_pix_op_new_8_1 ; exclude
10510 0000F2A8 8817                    <1>       mov    [edi], dl
10511 0000F2AA FF05[64030300]          <1>       inc    dword [u.r0]
10512                                  <1> m_pix_op_new_8_1:
10513 0000F2B0 47                      <1>       inc    edi
10514 0000F2B1 E2EC                    <1>       loop   m_pix_op_new_8_0
10515 0000F2B3 C3                      <1>       retn
10516                                  <1> m_pix_op_new_1:
10517 0000F2B4 803D[71120300]18        <1>       cmp    byte [v_bpp], 24 ; 24bpp
10518 0000F2BB 774B                    <1>       ja     short m_pix_op_new_3 ; 32bpp
10519 0000F2BD 722C                    <1>       jb     short m_pix_op_new_2 ; 16bpp
10520                                  <1>       ; 24 bit true colors
10521                                  <1>       ;jmp   short m_pix_op_new_24
10522                                  <1> m_pix_op_new_24:
10523                                  <1>       ; 24 bit true colors
10524 0000F2BF 89C2                    <1>       mov    edx, eax ; new color
10525                                  <1> m_pix_op_new_24_0:
10526 0000F2C1 66AD                    <1>       lodsw
10527 0000F2C3 C1E010                  <1>       shl    eax, 16
10528 0000F2C6 AC                      <1>       lodsb
10529 0000F2C7 C1C010                  <1>       rol    eax, 16
10530 0000F2CA 3B05[82120300]          <1>       cmp    eax, [maskcolor]
10531 0000F2D0 7413                    <1>       je     short m_pix_op_new_24_1 ; exclude
10532 0000F2D2 668917                  <1>       mov    [edi], dx
10533 0000F2D5 C1CA10                  <1>       ror    edx, 16
10534 0000F2D8 885702                  <1>       mov    [edi+2], dl
10535 0000F2DB C1C210                  <1>       rol    edx, 16
10536 0000F2DE 8305[64030300]03        <1>       add    dword [u.r0], 3
10537                                  <1> m_pix_op_new_24_1:
10538 0000F2E5 83C703                  <1>       add    edi, 3
10539 0000F2E8 E2D7                    <1>       loop   m_pix_op_new_24_0
10540 0000F2EA C3                      <1>       retn
10541                                  <1>       ; 65536 colors (16bpp)
10542                                  <1> m_pix_op_new_2:
10543                                  <1>       ;jmp   short m_pix_op_new_16
10544                                  <1> m_pix_op_new_16:
10545                                  <1>       ; 16 bit colors (65536 colors)
10546 0000F2EB 89C2                    <1>       mov    edx, eax ; new color
10547                                  <1> m_pix_op_new_16_0:
10548 0000F2ED 66AD                    <1>       lodsw
10549 0000F2EF 663B05[82120300]        <1>       cmp    ax, [maskcolor]
10550 0000F2F6 740A                    <1>       je     short m_pix_op_new_16_1 ; exclude
10551 0000F2F8 668917                  <1>       mov    [edi], dx
10552 0000F2FB 8305[64030300]02        <1>       add    dword [u.r0], 2
10553                                  <1> m_pix_op_new_16_1:
10554 0000F302 83C702                  <1>       add    edi, 2
10555 0000F305 E2E6                    <1>       loop   m_pix_op_new_16_0
10556 0000F307 C3                      <1>       retn
10557                                  <1> m_pix_op_new_3:
10558                                  <1>       ; 32 bit true colors
10559                                  <1>       ;jmp   short m_pix_op_new_32
10560                                  <1> m_pix_op_new_32:
10561                                  <1>       ; 32 bit true colors
10562 0000F308 89C2                    <1>       mov    edx, eax ; new color
10563                                  <1> m_pix_op_new_32_0:
10564 0000F30A AD                      <1>       lodsd
10565 0000F30B 3B05[82120300]          <1>       cmp    eax, [maskcolor]
10566 0000F311 7409                    <1>       je     short m_pix_op_new_32_1 ; exclude
10567 0000F313 8917                    <1>       mov    [edi], edx
10568 0000F315 8305[64030300]04        <1>       add    dword [u.r0], 4
10569                                  <1> m_pix_op_new_32_1:
10570 0000F31C 83C704                  <1>       add    edi, 4
10571 0000F31F E2E9                    <1>       loop   m_pix_op_new_32_0
10572 0000F321 C3                      <1>       retn
10573                                  <1>
10574                                  <1> m_pix_op_new_w:
10575                                  <1>       ; 06/02/2021
10576                                  <1>       ; CHANGE COLOR (MASKED, window)
10577                                  <1>       ;
10578                                  <1>       ; jump from pix_op_new_w
10579                                  <1>       ;
10580                                  <1>       ; INPUT:
10581                                  <1>       ;   ecx = bytes per row (to be applied)
10582                                  <1>       ;   edx = screen width in bytes
10583                                  <1>       ;   ebx = row count
10584                                  <1>       ;   eax = color
10585                                  <1>       ;
10586                                  <1>       ;   [maskcolor] = mask color (to be excluded)
10587                                  <1>       ;
10588                                  <1>       ; OUTPUT:
10589                                  <1>       ;      [u.r0] will be > 0 if succesful
10590                                  <1>
10591                                  <1>       ; Window
10592                                  <1>       ;mov   edi, [v_str] ; LFB start address
10593                                  <1>       ;mov   esi, edi
10594                                  <1>
10595 0000F322 803D[71120300]08        <1>       cmp    byte [v_bpp], 8 ; 8bpp
10596 0000F329 7707                    <1>       ja     short m_pix_op_new_w_1
10597                                  <1>
10598                                  <1>       ; 256 colors (8bpp)
10599 0000F32B BD[9DF20000]            <1>       mov    ebp, m_pix_op_new_8
10600 0000F330 EB1E                    <1>       jmp    short m_pix_op_new_w_x
10601                                  <1>
10602                                  <1> m_pix_op_new_w_1:
10603 0000F332 803D[71120300]18        <1>       cmp    byte [v_bpp], 24 ; 24bpp
10604 0000F339 7710                    <1>       ja     short m_pix_op_new_w_3 ; 32bpp
10605 0000F33B 7207                    <1>       jb     short m_pix_op_new_w_2 ; 16bpp
10606                                  <1>
10607                                  <1>       ; 24 bit true colors
10608 0000F33D BD[BFF20000]            <1>       mov    ebp, m_pix_op_new_24
```

```
10609 0000F342 EB0C                <1>       jmp    short m_pix_op_new_w_x
10610                              <1>
10611                              <1>       ; 65536 colors (16bpp)
10612                              <1> m_pix_op_new_w_2:
10613 0000F344 BD[EBF20000]        <1>       mov    ebp, m_pix_op_new_16
10614 0000F349 EB05                <1>       jmp    short m_pix_op_new_w_x
10615                              <1>
10616                              <1>       ; 32 bit true colors
10617                              <1> m_pix_op_new_w_3:
10618 0000F34B BD[08F30000]        <1>       mov    ebp, m_pix_op_new_32
10619                              <1>       ;jmp   short m_pix_op_new_w_x
10620                              <1>
10621                              <1> m_pix_op_new_w_x:
10622                              <1> m_pix_op_add_w_x:
10623                              <1> m_pix_op_sub_w_x:
10624                              <1> m_pix_op_mix_w_x:
10625                              <1> m_pix_op_and_w_x:
10626                              <1> m_pix_op_orc_w_x:
10627                              <1> m_pix_op_xor_w_x:
10628                              <1> m_pix_op_not_w_x:
10629                              <1> m_pix_op_neg_w_x:
10630                              <1> m_pix_op_inc_w_x:
10631                              <1> m_pix_op_dec_w_x:
10632                              <1>       ; 27/02/2021
10633                              <1>       ; 26/02/2021
10634                              <1>       ; 06/02/2021
10635                              <1>       ; ecx = bytes per row (to be applied)
10636                              <1>       ; edx = windows (screen) width in bytes
10637                              <1>       ; ebx = row count
10638                              <1>       ; eax = color
10639                              <1>       ; ebp = pixel operation subroutine address
10640                              <1>       ; edi = esi = window start address
10641                              <1>
10642 0000F350 8B3D[7A120300]      <1>       mov    edi, [v_str] ; LFB start address
10643 0000F356 89FE                <1>       mov    esi, edi
10644                              <1> m_pix_op_w_x_next:
10645 0000F358 52                  <1>       push   edx
10646 0000F359 51                  <1>       push   ecx
10647 0000F35A 56                  <1>       push   esi
10648 0000F35B 57                  <1>       push   edi
10649 0000F35C 50                  <1>       push   eax ; 26/02/2021
10650 0000F35D 8B0D[86120300]      <1>       mov    ecx, [pixcount] ; 27/02/2021
10651 0000F363 FFD5                <1>       call   ebp ; call masked pixel-row operation
10652 0000F365 58                  <1>       pop    eax ; 26/02/2021
10653 0000F366 5F                  <1>       pop    edi
10654 0000F367 5E                  <1>       pop    esi
10655 0000F368 59                  <1>       pop    ecx
10656 0000F369 5A                  <1>       pop    edx
10657 0000F36A 01D6                <1>       add    esi, edx ; next row
10658 0000F36C 01D7                <1>       add    edi, edx ; next row
10659 0000F36E 4B                  <1>       dec    ebx
10660 0000F36F 75E7                <1>       jnz    short m_pix_op_w_x_next
10661 0000F371 C3                  <1>       retn
10662                              <1>
10663                              <1> m_pix_op_add:
10664                              <1>       ; 06/02/2021
10665                              <1>       ; ADD COLOR (MASKED, full screen)
10666                              <1>       ;
10667                              <1>       ; jump from pix_op_add
10668                              <1>       ;
10669                              <1>       ; INPUT:
10670                              <1>       ;   eax = color (AL, AX, EAX)
10671                              <1>       ;   ecx = [v_siz] ; display page pixel count
10672                              <1>       ;   esi = edi = [v_mem] ; LFB start address
10673                              <1>       ;
10674                              <1>       ;   [maskcolor] = mask color (to be excluded)
10675                              <1>       ;
10676                              <1>       ; OUTPUT:
10677                              <1>       ;      [u.r0] will be > 0 if succesful
10678                              <1>
10679                              <1>       ; Full screen
10680                              <1> m_pix_op_add_0:
10681 0000F372 803D[71120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
10682 0000F379 771C                <1>       ja     short m_pix_op_add_1
10683                              <1>       ; 256 colors (8bpp)
10684                              <1>       ;jmp   short m_pix_op_add_8
10685                              <1> m_pix_op_add_8:
10686                              <1>       ; 8 bit colors (256 colors)
10687 0000F37B 88C2                <1>       mov    dl, al ; new color
10688                              <1> m_pix_op_add_8_0:
10689 0000F37D AC                  <1>       lodsb
10690 0000F37E 3A05[82120300]      <1>       cmp    al, [maskcolor]
10691 0000F384 740D                <1>       je     short m_pix_op_add_8_1 ; exclude
10692 0000F386 FF05[64030300]      <1>       inc    dword [u.r0] ; +1
10693 0000F38C 0017                <1>       add    [edi], dl
10694 0000F38E 7303                <1>       jnc    short m_pix_op_add_8_1
10695 0000F390 C607FF              <1>       mov    byte [edi], 0FFh
10696                              <1> m_pix_op_add_8_1:
10697 0000F393 47                  <1>       inc    edi
10698 0000F394 E2E7                <1>       loop   m_pix_op_add_8_0
10699 0000F396 C3                  <1>       retn
10700                              <1> m_pix_op_add_1:
10701 0000F397 803D[71120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
10702 0000F39E 775E                <1>       ja     short m_pix_op_add_3 ; 32bpp
10703 0000F3A0 7238                <1>       jb     short m_pix_op_add_2 ; 16bpp
10704                              <1>       ; 24 bit true colors
10705                              <1>       ;jmp   short m_pix_op_add_24
10706                              <1> m_pix_op_add_24:
10707                              <1>       ; 24 bit true colors
10708 0000F3A2 89C2                <1>       mov    edx, eax ; new color
10709 0000F3A4 81CA000000FF        <1>       or     edx, 0FF000000h
10710                              <1> m_pix_op_add_24_0:
10711 0000F3AA 66AD                <1>       lodsw
10712 0000F3AC C1E010              <1>       shl    eax, 16
10713 0000F3AF AC                  <1>       lodsb
```

```
10714 0000F3B0 C1C010              <1>       rol    eax, 16
10715 0000F3B3 3B05[82120300]      <1>       cmp    eax, [maskcolor]
10716 0000F3B9 7419                <1>       je     short m_pix_op_add_24_2 ; exclude
10717 0000F3BB 8305[64030300]03    <1>       add    dword [u.r0], 3 ; +3
10718 0000F3C2 01D0                <1>       add    eax, edx
10719 0000F3C4 7305                <1>       jnc    short m_pix_op_add_24_1
10720 0000F3C6 B8FFFFFF00          <1>       mov    eax, 0FFFFFFh
10721                              <1> m_pix_op_add_24_1:
10722 0000F3CB 668907              <1>       mov    [edi], ax
10723 0000F3CE C1E810              <1>       shr    eax, 16
10724 0000F3D1 884702              <1>       mov    [edi+2], al
10725                              <1> m_pix_op_add_24_2:
10726 0000F3D4 83C703              <1>       add    edi, 3 ; +3
10727 0000F3D7 E2D1                <1>       loop   m_pix_op_add_24_0
10728 0000F3D9 C3                  <1>       retn
10729                              <1>       ; 65536 colors (16bpp)
10730                              <1> m_pix_op_add_2:
10731                              <1>       ;jmp   short m_pix_op_add_16
10732                              <1> m_pix_op_add_16:
10733                              <1>       ; 16 bit colors (65536 colors)
10734 0000F3DA 89C2                <1>       mov    edx, eax ; new color
10735                              <1> m_pix_op_add_16_0:
10736 0000F3DC 66AD                <1>       lodsw
10737 0000F3DE 663B05[82120300]    <1>       cmp    ax, [maskcolor]
10738 0000F3E5 7411                <1>       je     short m_pix_op_add_16_1 ; exclude
10739 0000F3E7 8305[64030300]02    <1>       add    dword [u.r0], 2 ; +2
10740 0000F3EE 660117              <1>       add    [edi], dx
10741 0000F3F1 7305                <1>       jnc    short m_pix_op_add_16_1
10742 0000F3F3 66C707FFFF          <1>       mov    word [edi], 0FFFFh
10743                              <1> m_pix_op_add_16_1:
10744 0000F3F8 83C702              <1>       add    edi, 2 ; +2
10745 0000F3FB E2DF                <1>       loop   m_pix_op_add_16_0
10746 0000F3FD C3                  <1>       retn
10747                              <1> m_pix_op_add_3:
10748                              <1>       ; 32 bit true colors
10749                              <1>       ;jmp   short m_pix_op_add_32
10750                              <1> m_pix_op_add_32:
10751                              <1>       ; 32 bit true colors
10752 0000F3FE 89C2                <1>       mov    edx, eax ; new color
10753                              <1> m_pix_op_add_32_0:
10754 0000F400 AD                  <1>       lodsd
10755 0000F401 3B05[82120300]      <1>       cmp    eax, [maskcolor]
10756 0000F407 7411                <1>       je     short m_pix_op_add_32_1 ; exclude
10757 0000F409 8305[64030300]04    <1>       add    dword [u.r0], 4 ; +4
10758 0000F410 0117                <1>       add    [edi], edx
10759 0000F412 7306                <1>       jnc    short m_pix_op_add_32_1
10760 0000F414 C707FFFFFFFF        <1>       mov    dword [edi], 0FFFFFFFFh
10761                              <1> m_pix_op_add_32_1:
10762 0000F41A 83C704              <1>       add    edi, 4 ; +4
10763 0000F41D E2E1                <1>       loop   m_pix_op_add_32_0
10764 0000F41F C3                  <1>       retn
10765                              <1>
10766                              <1> m_pix_op_add_w:
10767                              <1>       ; 06/02/2021
10768                              <1>       ; ADD COLOR (MASKED, window)
10769                              <1>       ;
10770                              <1>       ; jump from pix_op_add_w
10771                              <1>       ;
10772                              <1>       ; INPUT:
10773                              <1>       ;   ecx = bytes per row (to be applied)
10774                              <1>       ;   edx = screen width in bytes
10775                              <1>       ;   ebx = row count
10776                              <1>       ;   eax = color
10777                              <1>       ;
10778                              <1>       ;   [maskcolor] = mask color (to be excluded)
10779                              <1>       ;
10780                              <1>       ; OUTPUT:
10781                              <1>       ;      [u.r0] will be > 0 if succesful
10782                              <1>
10783                              <1>       ; window
10784                              <1>       ;mov   edi, [v_str] ; LFB start address
10785                              <1>       ;mov   esi, edi
10786                              <1>
10787 0000F420 803D[71120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
10788 0000F427 7707                <1>       ja     short m_pix_op_add_w_1
10789                              <1>
10790                              <1>       ; 256 colors (8bpp)
10791 0000F429 BD[7BF30000]        <1>       mov    ebp, m_pix_op_add_8
10792 0000F42E EB1E                <1>       jmp    short m_pix_op_add_w_4
10793                              <1>
10794                              <1> m_pix_op_add_w_1:
10795 0000F430 803D[71120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
10796 0000F437 7710                <1>       ja     short m_pix_op_add_w_3 ; 32bpp
10797 0000F439 7207                <1>       jb     short m_pix_op_add_w_2 ; 16bpp
10798                              <1>
10799                              <1>       ; 24 bit true colors
10800 0000F43B BD[A2F30000]        <1>       mov    ebp, m_pix_op_add_24
10801 0000F440 EB0C                <1>       jmp    short m_pix_op_add_w_4
10802                              <1>
10803                              <1>       ; 65536 colors (16bpp)
10804                              <1> m_pix_op_add_w_2:
10805 0000F442 BD[DAF30000]        <1>       mov    ebp, m_pix_op_add_16
10806 0000F447 EB05                <1>       jmp    short m_pix_op_add_w_4
10807                              <1>
10808                              <1>       ; 32 bit true colors
10809                              <1> m_pix_op_add_w_3:
10810 0000F449 BD[FEF30000]        <1>       mov    ebp, m_pix_op_add_32
10811                              <1> m_pix_op_add_w_4:
10812 0000F44E E9FDFEFFFF          <1>       jmp    m_pix_op_add_w_x
10813                              <1>
10814                              <1> m_pix_op_sub:
10815                              <1>       ; 02/03/2021
10816                              <1>       ; 06/02/2021
10817                              <1>       ; SUBTRACT COLOR (MASKED, full screen)
10818                              <1>       ;
```

```
10819                                  <1>       ; jump from pix_op_sub
10820                                  <1>       ;
10821                                  <1>       ; INPUT:
10822                                  <1>       ;   eax = color (AL, AX, EAX)
10823                                  <1>       ;   ecx = [v_siz] ; display page pixel count
10824                                  <1>       ;   esi = edi = [v_mem] ; LFB start address
10825                                  <1>       ;
10826                                  <1>       ;   [maskcolor] = mask color (to be excluded)
10827                                  <1>       ;
10828                                  <1>       ; OUTPUT:
10829                                  <1>       ;      [u.r0] will be > 0 if succesful
10830                                  <1>
10831                                  <1>       ; Full screen
10832                                  <1> m_pix_op_sub_0:
10833 0000F453 803D[71120300]08        <1>       cmp    byte [v_bpp], 8 ; 8bpp
10834 0000F45A 771C                    <1>       ja     short m_pix_op_sub_1
10835                                  <1>       ; 256 colors (8bpp)
10836                                  <1>       ;jmp   short m_pix_op_sub_8
10837                                  <1> m_pix_op_sub_8:
10838                                  <1>       ; 8 bit colors (256 colors)
10839 0000F45C 88C2                    <1>       mov    dl, al ; new color
10840                                  <1> m_pix_op_sub_8_0:
10841 0000F45E AC                      <1>       lodsb
10842 0000F45F 3A05[82120300]          <1>       cmp    al, [maskcolor]
10843 0000F465 740D                    <1>       je     short m_pix_op_sub_8_1 ; exclude
10844 0000F467 FF05[64030300]          <1>       inc    dword [u.r0] ; +1
10845 0000F46D 2817                    <1>       sub    [edi], dl
10846 0000F46F 7303                    <1>       jnb    short m_pix_op_sub_8_1
10847 0000F471 C60700                  <1>       mov    byte [edi], 0
10848                                  <1> m_pix_op_sub_8_1:
10849 0000F474 47                      <1>       inc    edi
10850 0000F475 E2E7                    <1>       loop   m_pix_op_sub_8_0
10851 0000F477 C3                      <1>       retn
10852                                  <1> m_pix_op_sub_1:
10853 0000F478 803D[71120300]18        <1>       cmp    byte [v_bpp], 24 ; 24bpp
10854 0000F47F 7755                    <1>       ja     short m_pix_op_sub_3 ; 32bpp
10855 0000F481 722F                    <1>       jb     short m_pix_op_sub_2 ; 16bpp
10856                                  <1>       ; 24 bit true colors
10857                                  <1>       ;jmp   short m_pix_op_sub_24
10858                                  <1> m_pix_op_sub_24:
10859                                  <1>       ; 24 bit true colors
10860 0000F483 89C2                    <1>       mov    edx, eax ; new color
10861                                  <1>       ; 02/03/2021
10862                                  <1> m_pix_op_sub_24_0:
10863 0000F485 66AD                    <1>       lodsw
10864 0000F487 C1E010                  <1>       shl    eax, 16
10865 0000F48A AC                      <1>       lodsb
10866 0000F48B C1C010                  <1>       rol    eax, 16
10867 0000F48E 3B05[82120300]          <1>       cmp    eax, [maskcolor]
10868 0000F494 7416                    <1>       je     short m_pix_op_sub_24_2 ; exclude
10869 0000F496 8305[64030300]03        <1>       add    dword [u.r0], 3 ; +3
10870 0000F49D 29D0                    <1>       sub    eax, edx
10871 0000F49F 7302                    <1>       jnb    short m_pix_op_sub_24_1
10872 0000F4A1 31C0                    <1>       xor    eax, eax ; 0
10873                                  <1> m_pix_op_sub_24_1:
10874 0000F4A3 668907                  <1>       mov    [edi], ax
10875 0000F4A6 C1E810                  <1>       shr    eax, 16
10876 0000F4A9 884702                  <1>       mov    [edi+2], al
10877                                  <1> m_pix_op_sub_24_2:
10878 0000F4AC 83C703                  <1>       add    edi, 3 ; +3
10879 0000F4AF E2D4                    <1>       loop   m_pix_op_sub_24_0
10880 0000F4B1 C3                      <1>       retn
10881                                  <1>       ; 65536 colors (16bpp)
10882                                  <1> m_pix_op_sub_2:
10883                                  <1>       ;jmp   short m_pix_op_sub_16
10884                                  <1> m_pix_op_sub_16:
10885                                  <1>       ; 16 bit colors (65536 colors)
10886 0000F4B2 89C2                    <1>       mov    edx, eax ; new color
10887                                  <1> m_pix_op_sub_16_0:
10888 0000F4B4 66AD                    <1>       lodsw
10889 0000F4B6 663B05[82120300]        <1>       cmp    ax, [maskcolor]
10890 0000F4BD 7411                    <1>       je     short m_pix_op_sub_16_1 ; exclude
10891 0000F4BF 8305[64030300]02        <1>       add    dword [u.r0], 2 ; +2
10892 0000F4C6 662917                  <1>       sub    [edi], dx
10893 0000F4C9 7305                    <1>       jnb    short m_pix_op_sub_16_1
10894 0000F4CB 31C0                    <1>       xor    eax, eax
10895 0000F4CD 668907                  <1>       mov    [edi], ax ; 0
10896                                  <1> m_pix_op_sub_16_1:
10897 0000F4D0 83C702                  <1>       add    edi, 2 ; +2
10898 0000F4D3 E2DF                    <1>       loop   m_pix_op_sub_16_0
10899 0000F4D5 C3                      <1>       retn
10900                                  <1> m_pix_op_sub_3:
10901                                  <1>       ; 32 bit true colors
10902                                  <1>       ;jmp   short m_pix_op_sub_32
10903                                  <1> m_pix_op_sub_32:
10904                                  <1>       ; 32 bit true colors
10905 0000F4D6 89C2                    <1>       mov    edx, eax ; new color
10906                                  <1> m_pix_op_sub_32_0:
10907 0000F4D8 AD                      <1>       lodsd
10908 0000F4D9 3B05[82120300]          <1>       cmp    eax, [maskcolor]
10909 0000F4DF 740F                    <1>       je     short m_pix_op_sub_32_1 ; exclude
10910 0000F4E1 8305[64030300]04        <1>       add    dword [u.r0], 4 ; +4
10911 0000F4E8 2917                    <1>       sub    [edi], edx
10912 0000F4EA 7304                    <1>       jnb    short m_pix_op_sub_32_1
10913 0000F4EC 31C0                    <1>       xor    eax, eax
10914 0000F4EE 8907                    <1>       mov    [edi], eax ; 0
10915                                  <1> m_pix_op_sub_32_1:
10916 0000F4F0 83C704                  <1>       add    edi, 4 ; +4
10917 0000F4F3 E2E3                    <1>       loop   m_pix_op_sub_32_0
10918 0000F4F5 C3                      <1>       retn
10919                                  <1>
10920                                  <1> m_pix_op_sub_w:
10921                                  <1>       ; 06/02/2021
10922                                  <1>       ; SUBTRACT COLOR (MASKED, window)
10923                                  <1>       ;
```

```
10924                                  <1>       ; jump from pix_op_sub_w
10925                                  <1>       ;
10926                                  <1>       ; INPUT:
10927                                  <1>       ;   ecx = bytes per row (to be applied)
10928                                  <1>       ;   edx = screen width in bytes
10929                                  <1>       ;   ebx = row count
10930                                  <1>       ;   eax = color
10931                                  <1>       ;
10932                                  <1>       ;   [maskcolor] = mask color (to be excluded)
10933                                  <1>       ;
10934                                  <1>       ; OUTPUT:
10935                                  <1>       ;      [u.r0] will be > 0 if succesful
10936                                  <1>
10937                                  <1>       ; window
10938                                  <1>       ;mov   edi, [v_str] ; LFB start address
10939                                  <1>       ;mov   esi, edi
10940                                  <1>
10941 0000F4F6 803D[71120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
10942 0000F4FD 7707                   <1>       ja     short m_pix_op_sub_w_1
10943                                  <1>
10944                                  <1>       ; 256 colors (8bpp)
10945 0000F4FF BD[5CF40000]           <1>       mov    ebp, m_pix_op_sub_8
10946 0000F504 EB1E                   <1>       jmp    short m_pix_op_sub_w_4
10947                                  <1>
10948                                  <1> m_pix_op_sub_w_1:
10949 0000F506 803D[71120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
10950 0000F50D 7710                   <1>       ja     short m_pix_op_sub_w_3 ; 32bpp
10951 0000F50F 7207                   <1>       jb     short m_pix_op_sub_w_2 ; 16bpp
10952                                  <1>
10953                                  <1>       ; 24 bit true colors
10954 0000F511 BD[83F40000]           <1>       mov    ebp, m_pix_op_sub_24
10955 0000F516 EB0C                   <1>       jmp    short m_pix_op_sub_w_4
10956                                  <1>
10957                                  <1>       ; 65536 colors (16bpp)
10958                                  <1> m_pix_op_sub_w_2:
10959 0000F518 BD[B2F40000]           <1>       mov    ebp, m_pix_op_sub_16
10960 0000F51D EB05                   <1>       jmp    short m_pix_op_sub_w_4
10961                                  <1>
10962                                  <1>       ; 32 bit true colors
10963                                  <1> m_pix_op_sub_w_3:
10964 0000F51F BD[D6F40000]           <1>       mov    ebp, m_pix_op_sub_32
10965                                  <1> m_pix_op_sub_w_4:
10966 0000F524 E927FEFFFF             <1>       jmp    m_pix_op_sub_w_x
10967                                  <1>
10968                                  <1> m_pix_op_mix:
10969                                  <1>       ; 25/02/2021
10970                                  <1>       ; 06/02/2021
10971                                  <1>       ; MIX COLOR (MASKED, full screen)
10972                                  <1>       ;
10973                                  <1>       ; jump from pix_op_mix
10974                                  <1>       ;
10975                                  <1>       ; INPUT:
10976                                  <1>       ;   eax = color (AL, AX, EAX)
10977                                  <1>       ;   ecx = [v_siz] ; display page pixel count
10978                                  <1>       ;   esi = edi = [v_mem] ; LFB start address
10979                                  <1>       ;
10980                                  <1>       ;   [maskcolor] = mask color (to be excluded)
10981                                  <1>       ;
10982                                  <1>       ; OUTPUT:
10983                                  <1>       ;      [u.r0] will be > 0 if succesful
10984                                  <1>
10985                                  <1>       ; Full screen
10986                                  <1> m_pix_op_mix_0:
10987 0000F529 803D[71120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
10988 0000F530 771B                   <1>       ja     short m_pix_op_mix_1
10989                                  <1>       ; 256 colors (8bpp)
10990                                  <1>       ;jmp   short m_pix_op_mix_8
10991                                  <1> m_pix_op_mix_8:
10992                                  <1>       ; 8 bit colors (256 colors)
10993 0000F532 88C2                   <1>       mov    dl, al ; new (mixing) color
10994                                  <1> m_pix_op_mix_8_0:
10995 0000F534 AC                     <1>       lodsb
10996 0000F535 3A05[82120300]         <1>       cmp    al, [maskcolor]
10997 0000F53B 740C                   <1>       je     short m_pix_op_mix_8_1 ; exclude
10998 0000F53D 00D0                   <1>       add    al, dl ; 25/02/2021
10999 0000F53F D0D8                   <1>       rcr    al, 1
11000 0000F541 8807                   <1>       mov    [edi], al
11001 0000F543 FF05[64030300]         <1>       inc    dword [u.r0] ; +1
11002                                  <1> m_pix_op_mix_8_1:
11003 0000F549 47                     <1>       inc    edi
11004 0000F54A E2E8                   <1>       loop   m_pix_op_mix_8_0
11005 0000F54C C3                     <1>       retn
11006                                  <1> m_pix_op_mix_1:
11007 0000F54D 803D[71120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
11008 0000F554 7752                   <1>       ja     short m_pix_op_mix_3 ; 32bpp
11009 0000F556 722D                   <1>       jb     short m_pix_op_mix_2 ; 16bpp
11010                                  <1>       ; 24 bit true colors
11011                                  <1>       ;jmp   short m_pix_op_mix_24
11012                                  <1> m_pix_op_mix_24:
11013                                  <1>       ; 24 bit true colors
11014 0000F558 89C2                   <1>       mov    edx, eax ; new color
11015                                  <1>       ;and   edx, 0FFFFFFh
11016                                  <1> m_pix_op_mix_24_0:
11017 0000F55A 66AD                   <1>       lodsw
11018 0000F55C C1E010                 <1>       shl    eax, 16
11019 0000F55F AC                     <1>       lodsb
11020 0000F560 C1C010                 <1>       rol    eax, 16
11021 0000F563 3B05[82120300]         <1>       cmp    eax, [maskcolor]
11022 0000F569 7414                   <1>       je     short m_pix_op_mix_24_1 ; exclude
11023 0000F56B 01D0                   <1>       add    eax, edx
11024 0000F56D D1E8                   <1>       shr    eax, 1
11025 0000F56F 668907                 <1>       mov    [edi], ax
11026 0000F572 C1E810                 <1>       shr    eax, 16
11027 0000F575 884702                 <1>       mov    [edi+2], al
11028 0000F578 8305[64030300]03       <1>       add    dword [u.r0], 3 ; +3
```

```
11029                                  <1> m_pix_op_mix_24_1:
11030 0000F57F 83C703                  <1>         add    edi, 3 ; +3
11031 0000F582 E2D6                    <1>         loop   m_pix_op_mix_24_0
11032 0000F584 C3                      <1>         retn
11033                                  <1>         ; 65536 colors (16bpp)
11034                                  <1> m_pix_op_mix_2:
11035                                  <1>         ;jmp   short m_pix_op_mix_16
11036                                  <1> m_pix_op_mix_16:
11037                                  <1>         ; 16 bit colors (65536 colors)
11038 0000F585 89C2                    <1>         mov    edx, eax ; new color
11039                                  <1>         ;and   edx, 0FFFFh
11040                                  <1> m_pix_op_mix_16_0:
11041 0000F587 66AD                    <1>         lodsw
11042 0000F589 663B05[82120300]        <1>         cmp    ax, [maskcolor]
11043 0000F590 7410                    <1>         je     short m_pix_op_mix_16_1 ; exclude
11044 0000F592 6601D0                  <1>         add    ax, dx
11045 0000F595 66D1D8                  <1>         rcr    ax, 1
11046 0000F598 668907                  <1>         mov    [edi], ax
11047 0000F59B 8305[64030300]02        <1>         add    dword [u.r0], 2 ; +2
11048                                  <1> m_pix_op_mix_16_1:
11049 0000F5A2 83C702                  <1>         add    edi, 2 ; +2
11050 0000F5A5 E2E0                    <1>         loop   m_pix_op_mix_16_0
11051 0000F5A7 C3                      <1>         retn
11052                                  <1> m_pix_op_mix_3:
11053                                  <1>         ; 32 bit true colors
11054                                  <1>         ;jmp   short m_pix_op_mix_32
11055                                  <1> m_pix_op_mix_32:
11056                                  <1>         ; 32 bit true colors
11057 0000F5A8 89C2                    <1>         mov    edx, eax ; new color
11058                                  <1> m_pix_op_mix_32_0:
11059 0000F5AA AD                      <1>         lodsd
11060 0000F5AB 3B05[82120300]          <1>         cmp    eax, [maskcolor]
11061 0000F5B1 740D                    <1>         je     short m_pix_op_mix_32_1 ; exclude
11062 0000F5B3 01D0                    <1>         add    eax, edx
11063                                  <1>         ; 02/03/2021
11064 0000F5B5 D1D8                    <1>         rcr    eax, 1
11065 0000F5B7 8907                    <1>         mov    [edi], eax
11066 0000F5B9 8305[64030300]04        <1>         add    dword [u.r0], 4 ; +4
11067                                  <1> m_pix_op_mix_32_1:
11068 0000F5C0 83C704                  <1>         add    edi, 4 ; +4
11069 0000F5C3 E2E5                    <1>         loop   m_pix_op_mix_32_0
11070 0000F5C5 C3                      <1>         retn
11071                                  <1>
11072                                  <1> m_pix_op_mix_w:
11073                                  <1>         ; 06/02/2021
11074                                  <1>         ; MIX COLOR (MASKED, window)
11075                                  <1>         ;
11076                                  <1>         ; jump from pix_op_mix_w
11077                                  <1>         ;
11078                                  <1>         ; INPUT:
11079                                  <1>         ;   ecx = bytes per row (to be applied)
11080                                  <1>         ;   edx = screen width in bytes
11081                                  <1>         ;   ebx = row count
11082                                  <1>         ;   eax = color
11083                                  <1>         ;
11084                                  <1>         ;   [maskcolor] = mask color (to be excluded)
11085                                  <1>         ;
11086                                  <1>         ; OUTPUT:
11087                                  <1>         ;      [u.r0] will be > 0 if succesful
11088                                  <1>
11089                                  <1>         ; window
11090                                  <1>         ;mov   edi, [v_str] ; LFB start address
11091                                  <1>         ;mov   esi, edi
11092                                  <1>
11093 0000F5C6 803D[71120300]08        <1>         cmp    byte [v_bpp], 8 ; 8bpp
11094 0000F5CD 7707                    <1>         ja     short m_pix_op_mix_w_1
11095                                  <1>
11096                                  <1>         ; 256 colors (8bpp)
11097 0000F5CF BD[32F50000]            <1>         mov    ebp, m_pix_op_mix_8
11098 0000F5D4 EB1E                    <1>         jmp    short m_pix_op_mix_w_4
11099                                  <1>
11100                                  <1> m_pix_op_mix_w_1:
11101 0000F5D6 803D[71120300]18        <1>         cmp    byte [v_bpp], 24 ; 24bpp
11102 0000F5DD 7710                    <1>         ja     short m_pix_op_mix_w_3 ; 32bpp
11103 0000F5DF 7207                    <1>         jb     short m_pix_op_mix_w_2 ; 16bpp
11104                                  <1>
11105                                  <1>         ; 24 bit true colors
11106 0000F5E1 BD[58F50000]            <1>         mov    ebp, m_pix_op_mix_24
11107 0000F5E6 EB0C                    <1>         jmp    short m_pix_op_mix_w_4
11108                                  <1>
11109                                  <1>         ; 65536 colors (16bpp)
11110                                  <1> m_pix_op_mix_w_2:
11111 0000F5E8 BD[85F50000]            <1>         mov    ebp, m_pix_op_mix_16
11112 0000F5ED EB05                    <1>         jmp    short m_pix_op_mix_w_4
11113                                  <1>
11114                                  <1>         ; 32 bit true colors
11115                                  <1> m_pix_op_mix_w_3:
11116 0000F5EF BD[A8F50000]            <1>         mov    ebp, m_pix_op_mix_32
11117                                  <1> m_pix_op_mix_w_4:
11118 0000F5F4 E957FDFFFF              <1>         jmp    m_pix_op_mix_w_x
11119                                  <1>
11120                                  <1> m_pix_op_and:
11121                                  <1>         ; 06/02/2021
11122                                  <1>         ; AND COLOR (MASKED, full screen)
11123                                  <1>         ;
11124                                  <1>         ; jump from pix_op_and
11125                                  <1>         ;
11126                                  <1>         ; INPUT:
11127                                  <1>         ;   eax = color (AL, AX, EAX)
11128                                  <1>         ;   ecx = [v_siz] ; display page pixel count
11129                                  <1>         ;   esi = edi = [v_mem] ; LFB start address
11130                                  <1>         ;
11131                                  <1>         ;   [maskcolor] = mask color (to be excluded)
11132                                  <1>         ;
11133                                  <1>         ; OUTPUT:
```

```
11134                                  <1>       ;      [u.r0] will be > 0 if succesful
11135                                  <1>
11136                                  <1>       ; Full screen
11137                                  <1> m_pix_op_and_0:
11138 0000F5F9 803D[71120300]08        <1>       cmp    byte [v_bpp], 8 ; 8bpp
11139 0000F600 7717                    <1>       ja     short m_pix_op_and_1
11140                                  <1>       ; 256 colors (8bpp)
11141                                  <1>       ;jmp   short m_pix_op_and_8
11142                                  <1> m_pix_op_and_8:
11143                                  <1>       ; 8 bit colors (256 colors)
11144 0000F602 88C2                    <1>       mov    dl, al ; new color
11145                                  <1> m_pix_op_and_8_0:
11146 0000F604 AC                      <1>       lodsb
11147 0000F605 3A05[82120300]          <1>       cmp    al, [maskcolor]
11148 0000F60B 7408                    <1>       je     short m_pix_op_and_8_1 ; exclude
11149 0000F60D 2017                    <1>       and    [edi], dl
11150 0000F60F FF05[64030300]          <1>       inc    dword [u.r0] ; +1
11151                                  <1> m_pix_op_and_8_1:
11152 0000F615 47                      <1>       inc    edi
11153 0000F616 E2EC                    <1>       loop   m_pix_op_and_8_0
11154 0000F618 C3                      <1>       retn
11155                                  <1> m_pix_op_and_1:
11156 0000F619 803D[71120300]18        <1>       cmp    byte [v_bpp], 24 ; 24bpp
11157 0000F620 774A                    <1>       ja     short m_pix_op_and_3 ; 32bpp
11158 0000F622 722B                    <1>       jb     short m_pix_op_and_2 ; 16bpp
11159                                  <1>       ; 24 bit true colors
11160                                  <1>       ;jmp   short m_pix_op_and_24
11161                                  <1> m_pix_op_and_24:
11162                                  <1>       ; 24 bit true colors
11163 0000F624 89C2                    <1>       mov    edx, eax ; new color
11164                                  <1>       ;and   edx, 0FFFFFFh
11165                                  <1> m_pix_op_and_24_0:
11166 0000F626 66AD                    <1>       lodsw
11167 0000F628 C1E010                  <1>       shl    eax, 16
11168 0000F62B AC                      <1>       lodsb
11169 0000F62C C1C010                  <1>       rol    eax, 16
11170 0000F62F 3B05[82120300]          <1>       cmp    eax, [maskcolor]
11171 0000F635 7412                    <1>       je     short m_pix_op_and_24_1 ; exclude
11172 0000F637 21D0                    <1>       and    eax, edx
11173 0000F639 668907                  <1>       mov    [edi], ax
11174 0000F63C C1E810                  <1>       shr    eax, 16
11175 0000F63F 884702                  <1>       mov    [edi+2], al
11176 0000F642 8305[64030300]03        <1>       add    dword [u.r0], 3 ; +3
11177                                  <1> m_pix_op_and_24_1:
11178 0000F649 83C703                  <1>       add    edi, 3 ; +3
11179 0000F64C E2D8                    <1>       loop   m_pix_op_and_24_0
11180 0000F64E C3                      <1>       retn
11181                                  <1>       ; 65536 colors (16bpp)
11182                                  <1> m_pix_op_and_2:
11183                                  <1>       ;jmp   short m_pix_op_and_16
11184                                  <1> m_pix_op_and_16:
11185                                  <1>       ; 16 bit colors (65536 colors)
11186 0000F64F 89C2                    <1>       mov    edx, eax ; new color
11187                                  <1>       ;and   edx, 0FFFFh
11188                                  <1> m_pix_op_and_16_0:
11189 0000F651 66AD                    <1>       lodsw
11190 0000F653 663B05[82120300]        <1>       cmp    ax, [maskcolor]
11191 0000F65A 740A                    <1>       je     short m_pix_op_and_16_1 ; exclude
11192 0000F65C 662117                  <1>       and    [edi], dx
11193 0000F65F 8305[64030300]02        <1>       add    dword [u.r0], 2 ; +2
11194                                  <1> m_pix_op_and_16_1:
11195 0000F666 83C702                  <1>       add    edi, 2 ; +2
11196 0000F669 E2E6                    <1>       loop   m_pix_op_and_16_0
11197 0000F66B C3                      <1>       retn
11198                                  <1> m_pix_op_and_3:
11199                                  <1>       ; 32 bit true colors
11200                                  <1>       ;jmp   short m_pix_op_and_32
11201                                  <1> m_pix_op_and_32:
11202                                  <1>       ; 32 bit true colors
11203 0000F66C 89C2                    <1>       mov    edx, eax ; new color
11204                                  <1> m_pix_op_and_32_0:
11205 0000F66E AD                      <1>       lodsd
11206 0000F66F 3B05[82120300]          <1>       cmp    eax, [maskcolor]
11207 0000F675 7409                    <1>       je     short m_pix_op_and_32_1 ; exclude
11208 0000F677 2117                    <1>       and    [edi], edx ; 25/02/2021
11209 0000F679 8305[64030300]04        <1>       add    dword [u.r0], 4 ; +4
11210                                  <1> m_pix_op_and_32_1:
11211 0000F680 83C704                  <1>       add    edi, 4 ; +4
11212 0000F683 E2E9                    <1>       loop   m_pix_op_and_32_0
11213 0000F685 C3                      <1>       retn
11214                                  <1>
11215                                  <1> m_pix_op_and_w:
11216                                  <1>       ; 06/02/2021
11217                                  <1>       ; AND COLOR (MASKED, window)
11218                                  <1>       ;
11219                                  <1>       ; jump from pix_op_and_w
11220                                  <1>       ;
11221                                  <1>       ; INPUT:
11222                                  <1>       ;   ecx = bytes per row (to be applied)
11223                                  <1>       ;   edx = screen width in bytes
11224                                  <1>       ;   ebx = row count
11225                                  <1>       ;   eax = color
11226                                  <1>       ;
11227                                  <1>       ;   [maskcolor] = mask color (to be excluded)
11228                                  <1>       ;
11229                                  <1>       ; OUTPUT:
11230                                  <1>       ;      [u.r0] will be > 0 if succesful
11231                                  <1>
11232                                  <1>       ; window
11233                                  <1>       ;mov   edi, [v_str] ; LFB start address
11234                                  <1>       ;mov   esi, edi
11235                                  <1>
11236 0000F686 803D[71120300]08        <1>       cmp    byte [v_bpp], 8 ; 8bpp
11237 0000F68D 7707                    <1>       ja     short m_pix_op_and_w_1
11238                                  <1>
```

```
                                <1>       ; 256 colors (8bpp)
0000F68F BD[02F60000]           <1>       mov    ebp, m_pix_op_and_8
0000F694 EB1E                   <1>       jmp    short m_pix_op_and_w_4
                                <1>
                                <1> m_pix_op_and_w_1:
0000F696 803D[71120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000F69D 7710                   <1>       ja     short m_pix_op_and_w_3 ; 32bpp
0000F69F 7207                   <1>       jb     short m_pix_op_and_w_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0000F6A1 BD[24F60000]           <1>       mov    ebp, m_pix_op_and_24
0000F6A6 EB0C                   <1>       jmp    short m_pix_op_and_w_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> m_pix_op_and_w_2:
0000F6A8 BD[4FF60000]           <1>       mov    ebp, m_pix_op_and_16
0000F6AD EB05                   <1>       jmp    short m_pix_op_and_w_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> m_pix_op_and_w_3:
0000F6AF BD[6CF60000]           <1>       mov    ebp, m_pix_op_and_32
                                <1> m_pix_op_and_w_4:
0000F6B4 E997FCFFFF             <1>       jmp    m_pix_op_and_w_x
                                <1>
                                <1> m_pix_op_or:
                                <1>       ; 06/02/2021
                                <1>       ; OR COLOR (MASKED, full screen)
                                <1>       ;
                                <1>       ; jump from pix_op_orc
                                <1>       ;
                                <1>       ; INPUT:
                                <1>       ;   eax = color (AL, AX, EAX)
                                <1>       ;   ecx = [v_siz] ; display page pixel count
                                <1>       ;   esi = edi = [v_mem] ; LFB start address
                                <1>       ;
                                <1>       ;   [maskcolor] = mask color (to be excluded)
                                <1>       ;
                                <1>       ; OUTPUT:
                                <1>       ;      [u.r0] will be > 0 if succesful
                                <1>
                                <1>       ; Full screen
                                <1> m_pix_op_or_0:
0000F6B9 803D[71120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000F6C0 7717                   <1>       ja     short m_pix_op_or_1
                                <1>       ; 256 colors (8bpp)
                                <1>       ;jmp   short m_pix_op_or_8
                                <1> m_pix_op_or_8:
                                <1>       ; 8 bit colors (256 colors)
0000F6C2 88C2                   <1>       mov    dl, al ; new color
                                <1> m_pix_op_or_8_0:
0000F6C4 AC                     <1>       lodsb
0000F6C5 3A05[82120300]         <1>       cmp    al, [maskcolor]
0000F6CB 7408                   <1>       je     short m_pix_op_or_8_1 ; exclude
0000F6CD 0817                   <1>       or     [edi], dl
0000F6CF FF05[64030300]         <1>       inc    dword [u.r0] ; +1
                                <1> m_pix_op_or_8_1:
0000F6D5 47                     <1>       inc    edi
0000F6D6 E2EC                   <1>       loop   m_pix_op_or_8_0
0000F6D8 C3                     <1>       retn
                                <1> m_pix_op_or_1:
0000F6D9 803D[71120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000F6E0 774A                   <1>       ja     short m_pix_op_or_3 ; 32bpp
0000F6E2 722B                   <1>       jb     short m_pix_op_or_2 ; 16bpp
                                <1>       ; 24 bit true colors
                                <1>       ;jmp   short m_pix_op_or_24
                                <1> m_pix_op_or_24:
                                <1>       ; 24 bit true colors
0000F6E4 89C2                   <1>       mov    edx, eax ; new color
                                <1>       ;and   edx, 0FFFFFFh
                                <1> m_pix_op_or_24_0:
0000F6E6 66AD                   <1>       lodsw
0000F6E8 C1E010                 <1>       shl    eax, 16
0000F6EB AC                     <1>       lodsb
0000F6EC C1C010                 <1>       rol    eax, 16
0000F6EF 3B05[82120300]         <1>       cmp    eax, [maskcolor]
0000F6F5 7412                   <1>       je     short m_pix_op_or_24_1 ; exclude
0000F6F7 09D0                   <1>       or     eax, edx
0000F6F9 668907                 <1>       mov    [edi], ax
0000F6FC C1E810                 <1>       shr    eax, 16
0000F6FF 884702                 <1>       mov    [edi+2], al
0000F702 8305[64030300]03       <1>       add    dword [u.r0], 3 ; +3
                                <1> m_pix_op_or_24_1:
0000F709 83C703                 <1>       add    edi, 3 ; +3
0000F70C E2D8                   <1>       loop   m_pix_op_or_24_0
0000F70E C3                     <1>       retn
                                <1>       ; 65536 colors (16bpp)
                                <1> m_pix_op_or_2:
                                <1>       ;jmp   short m_pix_op_or_16
                                <1> m_pix_op_or_16:
                                <1>       ; 16 bit colors (65536 colors)
0000F70F 89C2                   <1>       mov    edx, eax ; new color
                                <1>       ;and   edx, 0FFFFh
                                <1> m_pix_op_or_16_0:
0000F711 66AD                   <1>       lodsw
0000F713 663B05[82120300]       <1>       cmp    ax, [maskcolor]
0000F71A 740A                   <1>       je     short m_pix_op_or_16_1 ; exclude
0000F71C 660917                 <1>       or     [edi], dx
0000F71F 8305[64030300]02       <1>       add    dword [u.r0], 2 ; +2
                                <1> m_pix_op_or_16_1:
0000F726 83C702                 <1>       add    edi, 2 ; +2
0000F729 E2E6                   <1>       loop   m_pix_op_or_16_0
0000F72B C3                     <1>       retn
                                <1> m_pix_op_or_3:
                                <1>       ; 32 bit true colors
                                <1>       ;jmp   short m_pix_op_or_32
```

```
11344                                  <1> m_pix_op_or_32:
11345                                  <1>       ; 32 bit true colors
11346 0000F72C 89C2                     <1>       mov    edx, eax ; new color
11347                                  <1> m_pix_op_or_32_0:
11348 0000F72E AD                       <1>       lodsd
11349 0000F72F 3B05[82120300]           <1>       cmp    eax, [maskcolor]
11350 0000F735 7409                     <1>       je     short m_pix_op_or_32_1 ; exclude
11351 0000F737 0917                     <1>       or     [edi], edx ; 25/02/2021
11352 0000F739 8305[64030300]04         <1>       add    dword [u.r0], 4 ; +4
11353                                  <1> m_pix_op_or_32_1:
11354 0000F740 83C704                   <1>       add    edi, 4 ; +4
11355 0000F743 E2E9                     <1>       loop   m_pix_op_or_32_0
11356 0000F745 C3                       <1>       retn
11357                                  <1>
11358                                  <1> m_pix_op_or_w:
11359                                  <1>       ; 06/02/2021
11360                                  <1>       ; MIX COLOR (MASKED, window)
11361                                  <1>       ;
11362                                  <1>       ; jump from pix_op_or_w
11363                                  <1>       ;
11364                                  <1>       ; INPUT:
11365                                  <1>       ;   ecx = bytes per row (to be applied)
11366                                  <1>       ;   edx = screen width in bytes
11367                                  <1>       ;   ebx = row count
11368                                  <1>       ;   eax = color
11369                                  <1>       ;
11370                                  <1>       ;   [maskcolor] = mask color (to be excluded)
11371                                  <1>       ;
11372                                  <1>       ; OUTPUT:
11373                                  <1>       ;      [u.r0] will be > 0 if succesful
11374                                  <1>
11375                                  <1>       ; window
11376                                  <1>       ;mov   edi, [v_str] ; LFB start address
11377                                  <1>       ;mov   esi, edi
11378                                  <1>
11379 0000F746 803D[71120300]08         <1>       cmp    byte [v_bpp], 8 ; 8bpp
11380 0000F74D 7707                     <1>       ja     short m_pix_op_or_w_1
11381                                  <1>
11382                                  <1>       ; 256 colors (8bpp)
11383 0000F74F BD[C2F60000]             <1>       mov    ebp, m_pix_op_or_8
11384 0000F754 EB1E                     <1>       jmp    short m_pix_op_or_w_4
11385                                  <1>
11386                                  <1> m_pix_op_or_w_1:
11387 0000F756 803D[71120300]18         <1>       cmp    byte [v_bpp], 24 ; 24bpp
11388 0000F75D 7710                     <1>       ja     short m_pix_op_or_w_3 ; 32bpp
11389 0000F75F 7207                     <1>       jb     short m_pix_op_or_w_2 ; 16bpp
11390                                  <1>
11391                                  <1>       ; 24 bit true colors
11392 0000F761 BD[E4F60000]             <1>       mov    ebp, m_pix_op_or_24
11393 0000F766 EB0C                     <1>       jmp    short m_pix_op_or_w_4
11394                                  <1>
11395                                  <1>       ; 65536 colors (16bpp)
11396                                  <1> m_pix_op_or_w_2:
11397 0000F768 BD[0FF70000]             <1>       mov    ebp, m_pix_op_or_16
11398 0000F76D EB05                     <1>       jmp    short m_pix_op_or_w_4
11399                                  <1>
11400                                  <1>       ; 32 bit true colors
11401                                  <1> m_pix_op_or_w_3:
11402 0000F76F BD[2CF70000]             <1>       mov    ebp, m_pix_op_or_32
11403                                  <1> m_pix_op_or_w_4:
11404 0000F774 E9D7FBFFFF               <1>       jmp    m_pix_op_orc_w_x
11405                                  <1>
11406                                  <1> m_pix_op_xor:
11407                                  <1>       ; 06/02/2021
11408                                  <1>       ; XOR COLOR (MASKED, full screen)
11409                                  <1>       ;
11410                                  <1>       ; jump from pix_op_xor
11411                                  <1>       ;
11412                                  <1>       ; INPUT:
11413                                  <1>       ;   eax = color (AL, AX, EAX)
11414                                  <1>       ;   ecx = [v_siz] ; display page pixel count
11415                                  <1>       ;   esi = edi = [v_mem] ; LFB start address
11416                                  <1>       ;
11417                                  <1>       ;   [maskcolor] = mask color (to be excluded)
11418                                  <1>       ;
11419                                  <1>       ; OUTPUT:
11420                                  <1>       ;      [u.r0] will be > 0 if succesful
11421                                  <1>
11422                                  <1>       ; Full screen
11423                                  <1> m_pix_op_xor_0:
11424 0000F779 803D[71120300]08         <1>       cmp    byte [v_bpp], 8 ; 8bpp
11425 0000F780 7717                     <1>       ja     short m_pix_op_xor_1
11426                                  <1>       ; 256 colors (8bpp)
11427                                  <1>       ;jmp   short m_pix_op_xor_8
11428                                  <1> m_pix_op_xor_8:
11429                                  <1>       ; 8 bit colors (256 colors)
11430 0000F782 88C2                     <1>       mov    dl, al ; new color
11431                                  <1> m_pix_op_xor_8_0:
11432 0000F784 AC                       <1>       lodsb
11433 0000F785 3A05[82120300]           <1>       cmp    al, [maskcolor]
11434 0000F78B 7408                     <1>       je     short m_pix_op_xor_8_1 ; exclude
11435 0000F78D 3017                     <1>       xor    [edi], dl
11436 0000F78F FF05[64030300]           <1>       inc    dword [u.r0] ; +1
11437                                  <1> m_pix_op_xor_8_1:
11438 0000F795 47                       <1>       inc    edi
11439 0000F796 E2EC                     <1>       loop   m_pix_op_xor_8_0
11440 0000F798 C3                       <1>       retn
11441                                  <1> m_pix_op_xor_1:
11442 0000F799 803D[71120300]18         <1>       cmp    byte [v_bpp], 24 ; 24bpp
11443 0000F7A0 774A                     <1>       ja     short m_pix_op_xor_3 ; 32bpp
11444 0000F7A2 722B                     <1>       jb     short m_pix_op_xor_2 ; 16bpp
11445                                  <1>       ; 24 bit true colors
11446                                  <1>       ;jmp   short m_pix_op_xor_24
11447                                  <1> m_pix_op_xor_24:
11448                                  <1>       ; 24 bit true colors
```

```
11449 0000F7A4 89C2              <1>       mov   edx, eax ; new color
11450                            <1>       ;and  edx, 0FFFFFFh
11451                            <1> m_pix_op_xor_24_0:
11452 0000F7A6 66AD              <1>       lodsw
11453 0000F7A8 C1E010            <1>       shl   eax, 16
11454 0000F7AB AC                <1>       lodsb
11455 0000F7AC C1C010            <1>       rol   eax, 16
11456 0000F7AF 3B05[82120300]    <1>       cmp   eax, [maskcolor]
11457 0000F7B5 7412              <1>       je    short m_pix_op_xor_24_1 ; exclude
11458 0000F7B7 31D0              <1>       xor   eax, edx
11459 0000F7B9 668907            <1>       mov   [edi], ax
11460 0000F7BC C1E810            <1>       shr   eax, 16
11461 0000F7BF 884702            <1>       mov   [edi+2], al
11462 0000F7C2 8305[64030300]03  <1>       add   dword [u.r0], 3 ; +3
11463                            <1> m_pix_op_xor_24_1:
11464 0000F7C9 83C703            <1>       add   edi, 3 ; +3
11465 0000F7CC E2D8              <1>       loop  m_pix_op_xor_24_0
11466 0000F7CE C3                <1>       retn
11467                            <1>       ; 65536 colors (16bpp)
11468                            <1> m_pix_op_xor_2:
11469                            <1>       ;jmp  short m_pix_op_xor_16
11470                            <1> m_pix_op_xor_16:
11471                            <1>       ; 16 bit colors (65536 colors)
11472 0000F7CF 89C2              <1>       mov   edx, eax ; new color
11473                            <1>       ;and  edx, 0FFFFh
11474                            <1> m_pix_op_xor_16_0:
11475 0000F7D1 66AD              <1>       lodsw
11476 0000F7D3 663B05[82120300]  <1>       cmp   ax, [maskcolor]
11477 0000F7DA 740A              <1>       je    short m_pix_op_xor_16_1 ; exclude
11478 0000F7DC 663117            <1>       xor   [edi], dx
11479 0000F7DF 8305[64030300]02  <1>       add   dword [u.r0], 2 ; +2
11480                            <1> m_pix_op_xor_16_1:
11481 0000F7E6 83C702            <1>       add   edi, 2 ; +2
11482 0000F7E9 E2E6              <1>       loop  m_pix_op_xor_16_0
11483 0000F7EB C3                <1>       retn
11484                            <1> m_pix_op_xor_3:
11485                            <1>       ; 32 bit true colors
11486                            <1>       ;jmp  short m_pix_op_xor_32
11487                            <1> m_pix_op_xor_32:
11488                            <1>       ; 32 bit true colors
11489 0000F7EC 89C2              <1>       mov   edx, eax ; new color
11490                            <1> m_pix_op_xor_32_0:
11491 0000F7EE AD                <1>       lodsd
11492 0000F7EF 3B05[82120300]    <1>       cmp   eax, [maskcolor]
11493 0000F7F5 7409              <1>       je    short m_pix_op_xor_32_1 ; exclude
11494 0000F7F7 3117              <1>       xor   [edi], edx ; 25/02/2021
11495 0000F7F9 8305[64030300]04  <1>       add   dword [u.r0], 4 ; +4
11496                            <1> m_pix_op_xor_32_1:
11497 0000F800 83C704            <1>       add   edi, 4 ; +4
11498 0000F803 E2E9              <1>       loop  m_pix_op_xor_32_0
11499 0000F805 C3                <1>       retn
11500                            <1>
11501                            <1> m_pix_op_xor_w:
11502                            <1>       ; 06/02/2021
11503                            <1>       ; XOR COLOR (MASKED, window)
11504                            <1>       ;
11505                            <1>       ; jump from pix_op_xor_w
11506                            <1>       ;
11507                            <1>       ; INPUT:
11508                            <1>       ;   ecx = bytes per row (to be applied)
11509                            <1>       ;   edx = screen width in bytes
11510                            <1>       ;   ebx = row count
11511                            <1>       ;   eax = color
11512                            <1>       ;
11513                            <1>       ;   [maskcolor] = mask color (to be excluded)
11514                            <1>       ;
11515                            <1>       ; OUTPUT:
11516                            <1>       ;     [u.r0] will be > 0 if succesful
11517                            <1>
11518                            <1>       ; window
11519                            <1>       ;mov  edi, [v_str] ; LFB start address
11520                            <1>       ;mov  esi, edi
11521                            <1>
11522 0000F806 803D[71120300]08  <1>       cmp   byte [v_bpp], 8 ; 8bpp
11523 0000F80D 7707              <1>       ja    short m_pix_op_xor_w_1
11524                            <1>
11525                            <1>       ; 256 colors (8bpp)
11526 0000F80F BD[82F70000]      <1>       mov   ebp, m_pix_op_xor_8
11527 0000F814 EB1E              <1>       jmp   short m_pix_op_xor_w_4
11528                            <1>
11529                            <1> m_pix_op_xor_w_1:
11530 0000F816 803D[71120300]18  <1>       cmp   byte [v_bpp], 24 ; 24bpp
11531 0000F81D 7710              <1>       ja    short m_pix_op_xor_w_3 ; 32bpp
11532 0000F81F 7207              <1>       jb    short m_pix_op_xor_w_2 ; 16bpp
11533                            <1>
11534                            <1>       ; 24 bit true colors
11535 0000F821 BD[A4F70000]      <1>       mov   ebp, m_pix_op_xor_24
11536 0000F826 EB0C              <1>       jmp   short m_pix_op_xor_w_4
11537                            <1>
11538                            <1>       ; 65536 colors (16bpp)
11539                            <1> m_pix_op_xor_w_2:
11540 0000F828 BD[CFF70000]      <1>       mov   ebp, m_pix_op_xor_16
11541 0000F82D EB05              <1>       jmp   short m_pix_op_xor_w_4
11542                            <1>
11543                            <1>       ; 32 bit true colors
11544                            <1> m_pix_op_xor_w_3:
11545 0000F82F BD[ECF70000]      <1>       mov   ebp, m_pix_op_xor_32
11546                            <1> m_pix_op_xor_w_4:
11547 0000F834 E917FBFFFF        <1>       jmp   m_pix_op_xor_w_x
11548                            <1>
11549                            <1> m_pix_op_not:
11550                            <1>       ; 06/02/2021
11551                            <1>       ; NOT COLOR (MASKED, full screen)
11552                            <1>       ;
11553                            <1>       ; jump from pix_op_not
```

```
11554                                  <1>       ;
11555                                  <1>       ; INPUT:
11556                                  <1>       ;   ecx = [v_siz] ; display page pixel count
11557                                  <1>       ;   esi = edi = [v_mem] ; LFB start address
11558                                  <1>       ;
11559                                  <1>       ;   [maskcolor] = mask color (to be excluded)
11560                                  <1>       ;
11561                                  <1>       ; OUTPUT:
11562                                  <1>       ;      [u.r0] will be > 0 if succesful
11563                                  <1>
11564                                  <1>       ; Full screen
11565                                  <1> m_pix_op_not_0:
11566 0000F839 803D[71120300]08        <1>       cmp    byte [v_bpp], 8 ; 8bpp
11567 0000F840 7715                    <1>       ja     short m_pix_op_not_1
11568                                  <1>       ; 256 colors (8bpp)
11569                                  <1>       ;jmp   short m_pix_op_not_8
11570                                  <1> m_pix_op_not_8:
11571                                  <1>       ; 8 bit colors (256 colors)
11572 0000F842 AC                      <1>       lodsb
11573 0000F843 3A05[82120300]          <1>       cmp    al, [maskcolor]
11574 0000F849 7408                    <1>       je     short m_pix_op_not_8_1 ; exclude
11575 0000F84B F617                    <1>       not    byte [edi]
11576 0000F84D FF05[64030300]          <1>       inc    dword [u.r0] ; +1
11577                                  <1> m_pix_op_not_8_1:
11578 0000F853 47                      <1>       inc    edi
11579 0000F854 E2EC                    <1>       loop   m_pix_op_not_8
11580 0000F856 C3                      <1>       retn
11581                                  <1> m_pix_op_not_1:
11582 0000F857 803D[71120300]18        <1>       cmp    byte [v_bpp], 24 ; 24bpp
11583 0000F85E 7746                    <1>       ja     short m_pix_op_not_3 ; 32bpp
11584 0000F860 7229                    <1>       jb     short m_pix_op_not_2 ; 16bpp
11585                                  <1>       ; 24 bit true colors
11586                                  <1>       ;jmp   short m_pix_op_not_24
11587                                  <1> m_pix_op_not_24:
11588                                  <1>       ; 24 bit true colors
11589 0000F862 66AD                    <1>       lodsw
11590 0000F864 C1E010                  <1>       shl    eax, 16
11591 0000F867 AC                      <1>       lodsb
11592 0000F868 C1C010                  <1>       rol    eax, 16
11593 0000F86B 3B05[82120300]          <1>       cmp    eax, [maskcolor]
11594 0000F871 7412                    <1>       je     short m_pix_op_not_24_1 ; exclude
11595 0000F873 F7D0                    <1>       not    eax
11596 0000F875 668907                  <1>       mov    [edi], ax
11597 0000F878 C1E810                  <1>       shr    eax, 16
11598 0000F87B 884702                  <1>       mov    [edi+2], al
11599 0000F87E 8305[64030300]03        <1>       add    dword [u.r0], 3 ; +3
11600                                  <1> m_pix_op_not_24_1:
11601 0000F885 83C703                  <1>       add    edi, 3 ; +3
11602 0000F888 E2D8                    <1>       loop   m_pix_op_not_24
11603 0000F88A C3                      <1>       retn
11604                                  <1>       ; 65536 colors (16bpp)
11605                                  <1> m_pix_op_not_2:
11606                                  <1>       ;jmp   short m_pix_op_not_16
11607                                  <1> m_pix_op_not_16:
11608                                  <1>       ; 16 bit colors (65536 colors)
11609 0000F88B 66AD                    <1>       lodsw
11610 0000F88D 663B05[82120300]        <1>       cmp    ax, [maskcolor]
11611 0000F894 740A                    <1>       je     short m_pix_op_not_16_1 ; exclude
11612 0000F896 66F717                  <1>       not    word [edi]
11613 0000F899 8305[64030300]02        <1>       add    dword [u.r0], 2 ; +2
11614                                  <1> m_pix_op_not_16_1:
11615 0000F8A0 83C702                  <1>       add    edi, 2 ; +2
11616 0000F8A3 E2E6                    <1>       loop   m_pix_op_not_16
11617 0000F8A5 C3                      <1>       retn
11618                                  <1> m_pix_op_not_3:
11619                                  <1>       ; 32 bit true colors
11620                                  <1>       ;jmp   short m_pix_op_not_32
11621                                  <1> m_pix_op_not_32:
11622                                  <1>       ; 32 bit true colors
11623 0000F8A6 AD                      <1>       lodsd
11624 0000F8A7 3B05[82120300]          <1>       cmp    eax, [maskcolor]
11625 0000F8AD 7409                    <1>       je     short m_pix_op_not_32_1 ; exclude
11626 0000F8AF F717                    <1>       not    dword [edi]
11627 0000F8B1 8305[64030300]04        <1>       add    dword [u.r0], 4 ; +4
11628                                  <1> m_pix_op_not_32_1:
11629 0000F8B8 83C704                  <1>       add    edi, 4 ; +4
11630 0000F8BB E2E9                    <1>       loop   m_pix_op_not_32
11631 0000F8BD C3                      <1>       retn
11632                                  <1>
11633                                  <1> m_pix_op_not_w:
11634                                  <1>       ; 06/02/2021
11635                                  <1>       ; NOT COLOR (MASKED, window)
11636                                  <1>       ;
11637                                  <1>       ; jump from pix_op_not_w
11638                                  <1>       ;
11639                                  <1>       ; INPUT:
11640                                  <1>       ;   ecx = bytes per row (to be applied)
11641                                  <1>       ;   edx = screen width in bytes
11642                                  <1>       ;   ebx = row count
11643                                  <1>       ;
11644                                  <1>       ;   [maskcolor] = mask color (to be excluded)
11645                                  <1>       ;
11646                                  <1>       ; OUTPUT:
11647                                  <1>       ;      [u.r0] will be > 0 if succesful
11648                                  <1>
11649                                  <1>       ; window
11650                                  <1>       ;mov   edi, [v_str] ; LFB start address
11651                                  <1>       ;mov   esi, edi
11652                                  <1>
11653 0000F8BE 803D[71120300]08        <1>       cmp    byte [v_bpp], 8 ; 8bpp
11654 0000F8C5 7707                    <1>       ja     short m_pix_op_not_w_1
11655                                  <1>
11656                                  <1>       ; 256 colors (8bpp)
11657 0000F8C7 BD[42F80000]            <1>       mov    ebp, m_pix_op_not_8
11658 0000F8CC EB1E                    <1>       jmp    short m_pix_op_not_w_4
```

```
11659                                  <1>
11660                                  <1> m_pix_op_not_w_1:
11661 0000F8CE 803D[71120300]18          <1>       cmp    byte [v_bpp], 24 ; 24bpp
11662 0000F8D5 7710                      <1>       ja     short m_pix_op_not_w_3 ; 32bpp
11663 0000F8D7 7207                      <1>       jb     short m_pix_op_not_w_2 ; 16bpp
11664                                  <1>
11665                                  <1>       ; 24 bit true colors
11666 0000F8D9 BD[62F80000]              <1>       mov    ebp, m_pix_op_not_24
11667 0000F8DE EB0C                      <1>       jmp    short m_pix_op_not_w_4
11668                                  <1>
11669                                  <1>       ; 65536 colors (16bpp)
11670                                  <1> m_pix_op_not_w_2:
11671 0000F8E0 BD[8BF80000]              <1>       mov    ebp, m_pix_op_not_16
11672 0000F8E5 EB05                      <1>       jmp    short m_pix_op_not_w_4
11673                                  <1>
11674                                  <1>       ; 32 bit true colors
11675                                  <1> m_pix_op_not_w_3:
11676 0000F8E7 BD[A6F80000]              <1>       mov    ebp, m_pix_op_not_32
11677                                  <1> m_pix_op_not_w_4:
11678 0000F8EC E95FFAFFFF                <1>       jmp    m_pix_op_not_w_x
11679                                  <1>
11680                                  <1> m_pix_op_neg:
11681                                  <1>       ; 06/02/2021
11682                                  <1>       ; NEGATIVE COLOR (MASKED, full screen)
11683                                  <1>       ;
11684                                  <1>       ; jump from pix_op_neg
11685                                  <1>       ;
11686                                  <1>       ; INPUT:
11687                                  <1>       ;   ecx = [v_siz] ; display page pixel count
11688                                  <1>       ;   esi = edi = [v_mem] ; LFB start address
11689                                  <1>       ;
11690                                  <1>       ;   [maskcolor] = mask color (to be excluded)
11691                                  <1>       ;
11692                                  <1>       ; OUTPUT:
11693                                  <1>       ;      [u.r0] will be > 0 if succesful
11694                                  <1>
11695                                  <1>       ; Full screen
11696                                  <1> m_pix_op_neg_0:
11697 0000F8F1 803D[71120300]08          <1>       cmp    byte [v_bpp], 8 ; 8bpp
11698 0000F8F8 7715                      <1>       ja     short m_pix_op_neg_1
11699                                  <1>       ; 256 colors (8bpp)
11700                                  <1>       ;jmp   short m_pix_op_neg_8
11701                                  <1> m_pix_op_neg_8:
11702                                  <1>       ; 8 bit colors (256 colors)
11703 0000F8FA AC                        <1>       lodsb
11704 0000F8FB 3A05[82120300]            <1>       cmp    al, [maskcolor]
11705 0000F901 7408                      <1>       je     short m_pix_op_neg_8_1 ; exclude
11706 0000F903 F61F                      <1>       neg    byte [edi]
11707 0000F905 FF05[64030300]            <1>       inc    dword [u.r0] ; +1
11708                                  <1> m_pix_op_neg_8_1:
11709 0000F90B 47                        <1>       inc    edi
11710 0000F90C E2EC                      <1>       loop   m_pix_op_neg_8
11711 0000F90E C3                        <1>       retn
11712                                  <1> m_pix_op_neg_1:
11713 0000F90F 803D[71120300]18          <1>       cmp    byte [v_bpp], 24 ; 24bpp
11714 0000F916 7746                      <1>       ja     short m_pix_op_neg_3 ; 32bpp
11715 0000F918 7229                      <1>       jb     short m_pix_op_neg_2 ; 16bpp
11716                                  <1>       ; 24 bit true colors
11717                                  <1>       ;jmp   short m_pix_op_neg_24
11718                                  <1> m_pix_op_neg_24:
11719                                  <1>       ; 24 bit true colors
11720 0000F91A 66AD                      <1>       lodsw
11721 0000F91C C1E010                    <1>       shl    eax, 16
11722 0000F91F AC                        <1>       lodsb
11723 0000F920 C1C010                    <1>       rol    eax, 16
11724 0000F923 3B05[82120300]            <1>       cmp    eax, [maskcolor]
11725 0000F929 7412                      <1>       je     short m_pix_op_neg_24_1 ; exclude
11726 0000F92B F7D8                      <1>       neg    eax
11727 0000F92D 668907                    <1>       mov    [edi], ax
11728 0000F930 C1E810                    <1>       shr    eax, 16
11729 0000F933 884702                    <1>       mov    [edi+2], al
11730 0000F936 8305[64030300]03          <1>       add    dword [u.r0], 3 ; +3
11731                                  <1> m_pix_op_neg_24_1:
11732 0000F93D 83C703                    <1>       add    edi, 3 ; +3
11733 0000F940 E2D8                      <1>       loop   m_pix_op_neg_24
11734 0000F942 C3                        <1>       retn
11735                                  <1>       ; 65536 colors (16bpp)
11736                                  <1> m_pix_op_neg_2:
11737                                  <1>       ;jmp   short m_pix_op_neg_16
11738                                  <1> m_pix_op_neg_16:
11739                                  <1>       ; 16 bit colors (65536 colors)
11740 0000F943 66AD                      <1>       lodsw
11741 0000F945 663B05[82120300]          <1>       cmp    ax, [maskcolor]
11742 0000F94C 740A                      <1>       je     short m_pix_op_neg_16_1 ; exclude
11743 0000F94E 66F71F                    <1>       neg    word [edi]
11744 0000F951 8305[64030300]02          <1>       add    dword [u.r0], 2 ; +2
11745                                  <1> m_pix_op_neg_16_1:
11746 0000F958 83C702                    <1>       add    edi, 2 ; +2
11747 0000F95B E2E6                      <1>       loop   m_pix_op_neg_16
11748 0000F95D C3                        <1>       retn
11749                                  <1> m_pix_op_neg_3:
11750                                  <1>       ; 32 bit true colors
11751                                  <1>       ;jmp   short m_pix_op_neg_32
11752                                  <1> m_pix_op_neg_32:
11753                                  <1>       ; 32 bit true colors
11754 0000F95E AD                        <1>       lodsd
11755 0000F95F 3B05[82120300]            <1>       cmp    eax, [maskcolor]
11756 0000F965 7409                      <1>       je     short m_pix_op_neg_32_1 ; exclude
11757 0000F967 F71F                      <1>       neg    dword [edi]
11758 0000F969 8305[64030300]04          <1>       add    dword [u.r0], 4 ; +4
11759                                  <1> m_pix_op_neg_32_1:
11760 0000F970 83C704                    <1>       add    edi, 4 ; +4
11761 0000F973 E2E9                      <1>       loop   m_pix_op_neg_32
11762 0000F975 C3                        <1>       retn
11763                                  <1>
```

```
11764                                  <1> m_pix_op_neg_w:
11765                                  <1>       ; 06/02/2021
11766                                  <1>       ; NEGATIVE COLOR (MASKED, window)
11767                                  <1>       ;
11768                                  <1>       ; jump from pix_op_neg_w
11769                                  <1>       ;
11770                                  <1>       ; INPUT:
11771                                  <1>       ;   ecx = bytes per row (to be applied)
11772                                  <1>       ;   edx = screen width in bytes
11773                                  <1>       ;   ebx = row count
11774                                  <1>       ;
11775                                  <1>       ;   [maskcolor] = mask color (to be excluded)
11776                                  <1>       ;
11777                                  <1>       ; OUTPUT:
11778                                  <1>       ;     [u.r0] will be > 0 if succesful
11779                                  <1>
11780                                  <1>       ; window
11781                                  <1>       ;mov   edi, [v_str] ; LFB start address
11782                                  <1>       ;mov   esi, edi
11783                                  <1>
11784 0000F976 803D[71120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
11785 0000F97D 7707                   <1>       ja     short m_pix_op_neg_w_1
11786                                  <1>
11787                                  <1>       ; 256 colors (8bpp)
11788 0000F97F BD[FAF80000]           <1>       mov    ebp, m_pix_op_neg_8
11789 0000F984 EB1E                   <1>       jmp    short m_pix_op_neg_w_4
11790                                  <1>
11791                                  <1> m_pix_op_neg_w_1:
11792 0000F986 803D[71120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
11793 0000F98D 7710                   <1>       ja     short m_pix_op_neg_w_3 ; 32bpp
11794 0000F98F 7207                   <1>       jb     short m_pix_op_neg_w_2 ; 16bpp
11795                                  <1>
11796                                  <1>       ; 24 bit true colors
11797 0000F991 BD[1AF90000]           <1>       mov    ebp, m_pix_op_neg_24
11798 0000F996 EB0C                   <1>       jmp    short m_pix_op_neg_w_4
11799                                  <1>
11800                                  <1>       ; 65536 colors (16bpp)
11801                                  <1> m_pix_op_neg_w_2:
11802 0000F998 BD[43F90000]           <1>       mov    ebp, m_pix_op_neg_16
11803 0000F99D EB05                   <1>       jmp    short m_pix_op_neg_w_4
11804                                  <1>
11805                                  <1>       ; 32 bit true colors
11806                                  <1> m_pix_op_neg_w_3:
11807 0000F99F BD[5EF90000]           <1>       mov    ebp, m_pix_op_neg_32
11808                                  <1> m_pix_op_neg_w_4:
11809 0000F9A4 E9A7F9FFFF             <1>       jmp    m_pix_op_neg_w_x
11810                                  <1>
11811                                  <1> m_pix_op_inc:
11812                                  <1>       ; 06/02/2021
11813                                  <1>       ; INCREASE COLOR (MASKED, full screen)
11814                                  <1>       ;
11815                                  <1>       ; jump from pix_op_inc
11816                                  <1>       ;
11817                                  <1>       ; INPUT:
11818                                  <1>       ;   ecx = [v_siz] ; display page pixel count
11819                                  <1>       ;   esi = edi = [v_mem] ; LFB start address
11820                                  <1>       ;
11821                                  <1>       ;   [maskcolor] = mask color (to be excluded)
11822                                  <1>       ;
11823                                  <1>       ; OUTPUT:
11824                                  <1>       ;     [u.r0] will be > 0 if succesful
11825                                  <1>
11826                                  <1>       ; Full screen
11827                                  <1> m_pix_op_inc_0:
11828 0000F9A9 803D[71120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
11829 0000F9B0 7719                   <1>       ja     short m_pix_op_inc_1
11830                                  <1>       ; 256 colors (8bpp)
11831                                  <1>       ;jmp   short m_pix_op_inc_8
11832                                  <1> m_pix_op_inc_8:
11833                                  <1>       ; 8 bit colors (256 colors)
11834 0000F9B2 AC                     <1>       lodsb
11835 0000F9B3 3A05[82120300]         <1>       cmp    al, [maskcolor]
11836 0000F9B9 740C                   <1>       je     short m_pix_op_inc_8_1 ; exclude
11837 0000F9BB FE07                   <1>       inc    byte [edi]
11838 0000F9BD 7502                   <1>       jnz    short m_pix_op_inc_8_0
11839 0000F9BF FE0F                   <1>       dec    byte [edi]
11840                                  <1> m_pix_op_inc_8_0:
11841 0000F9C1 FF05[64030300]         <1>       inc    dword [u.r0] ; +1
11842                                  <1> m_pix_op_inc_8_1:
11843 0000F9C7 47                     <1>       inc    edi
11844 0000F9C8 E2E8                   <1>       loop   m_pix_op_inc_8
11845 0000F9CA C3                     <1>       retn
11846                                  <1> m_pix_op_inc_1:
11847 0000F9CB 803D[71120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
11848 0000F9D2 7752                   <1>       ja     short m_pix_op_inc_3 ; 32bpp
11849 0000F9D4 7230                   <1>       jb     short m_pix_op_inc_2 ; 16bpp
11850                                  <1>       ; 24 bit true colors
11851                                  <1>       ;jmp   short m_pix_op_inc_24
11852                                  <1> m_pix_op_inc_24:
11853                                  <1>       ; 24 bit true colors
11854 0000F9D6 66AD                   <1>       lodsw
11855 0000F9D8 C1E010                 <1>       shl    eax, 16
11856 0000F9DB AC                     <1>       lodsb
11857 0000F9DC C1C010                 <1>       rol    eax, 16
11858 0000F9DF 3B05[82120300]         <1>       cmp    eax, [maskcolor]
11859 0000F9E5 7419                   <1>       je     short m_pix_op_inc_24_1 ; exclude
11860 0000F9E7 40                     <1>       inc    eax
11861 0000F9E8 3DFFFFFF00             <1>       cmp    eax, 0FFFFFFh
11862 0000F9ED 7601                   <1>       jna    short m_pix_op_inc_24_0
11863 0000F9EF 48                     <1>       dec    eax
11864                                  <1> m_pix_op_inc_24_0:
11865 0000F9F0 668907                 <1>       mov    [edi], ax
11866 0000F9F3 C1E810                 <1>       shr    eax, 16
11867 0000F9F6 884702                 <1>       mov    [edi+2], al
11868 0000F9F9 8305[64030300]03       <1>       add    dword [u.r0], 3 ; +3
```

```
11869                                  <1> m_pix_op_inc_24_1:
11870 0000FA00 83C703                  <1>         add    edi, 3 ; +3
11871 0000FA03 E2D1                    <1>         loop   m_pix_op_inc_24
11872 0000FA05 C3                      <1>         retn
11873                                  <1>         ; 65536 colors (16bpp)
11874                                  <1> m_pix_op_inc_2:
11875                                  <1>         ;jmp   short m_pix_op_inc_16
11876                                  <1> m_pix_op_inc_16:
11877                                  <1>         ; 16 bit colors (65536 colors)
11878 0000FA06 66AD                    <1>         lodsw
11879 0000FA08 663B05[82120300]        <1>         cmp    ax, [maskcolor]
11880 0000FA0F 740F                    <1>         je     short m_pix_op_inc_16_1 ; exclude
11881 0000FA11 66FF07                  <1>         inc    word [edi]
11882 0000FA14 7503                    <1>         jnz    short m_pix_op_inc_16_0
11883 0000FA16 66FF0F                  <1>         dec    word [edi]
11884                                  <1> m_pix_op_inc_16_0:
11885 0000FA19 8305[64030300]02        <1>         add    dword [u.r0], 2 ; +2
11886                                  <1> m_pix_op_inc_16_1:
11887 0000FA20 83C702                  <1>         add    edi, 2 ; +2
11888 0000FA23 E2E1                    <1>         loop   m_pix_op_inc_16
11889 0000FA25 C3                      <1>         retn
11890                                  <1> m_pix_op_inc_3:
11891                                  <1>         ; 32 bit true colors
11892                                  <1>         ;jmp   short m_pix_op_inc_32
11893                                  <1> m_pix_op_inc_32:
11894                                  <1>         ; 32 bit true colors
11895 0000FA26 AD                      <1>         lodsd
11896 0000FA27 3B05[82120300]          <1>         cmp    eax, [maskcolor]
11897 0000FA2D 740D                    <1>         je     short m_pix_op_inc_32_1 ; exclude
11898 0000FA2F FF07                    <1>         inc    dword [edi]
11899 0000FA31 7502                    <1>         jnz    short m_pix_op_inc_32_0
11900 0000FA33 FF0F                    <1>         dec    dword [edi]
11901                                  <1> m_pix_op_inc_32_0:
11902 0000FA35 8305[64030300]04        <1>         add    dword [u.r0], 4 ; +4
11903                                  <1> m_pix_op_inc_32_1:
11904 0000FA3C 83C704                  <1>         add    edi, 4 ; +4
11905 0000FA3F E2E5                    <1>         loop   m_pix_op_inc_32
11906 0000FA41 C3                      <1>         retn
11907                                  <1>
11908                                  <1> m_pix_op_inc_w:
11909                                  <1>         ; 06/02/2021
11910                                  <1>         ; INCREASE COLOR (MASKED, window)
11911                                  <1>         ;
11912                                  <1>         ; jump from pix_op_inc_w
11913                                  <1>         ;
11914                                  <1>         ; INPUT:
11915                                  <1>         ;   ecx = bytes per row (to be applied)
11916                                  <1>         ;   edx = screen width in bytes
11917                                  <1>         ;   ebx = row count
11918                                  <1>         ;
11919                                  <1>         ;   [maskcolor] = mask color (to be excluded)
11920                                  <1>         ;
11921                                  <1>         ; OUTPUT:
11922                                  <1>         ;     [u.r0] will be > 0 if succesful
11923                                  <1>
11924                                  <1>         ; window
11925                                  <1>         ;mov   edi, [v_str] ; LFB start address
11926                                  <1>         ;mov   esi, edi
11927                                  <1>
11928 0000FA42 803D[71120300]08        <1>         cmp    byte [v_bpp], 8 ; 8bpp
11929 0000FA49 7707                    <1>         ja     short m_pix_op_inc_w_1
11930                                  <1>
11931                                  <1>         ; 256 colors (8bpp)
11932 0000FA4B BD[B2F90000]            <1>         mov    ebp, m_pix_op_inc_8
11933 0000FA50 EB1E                    <1>         jmp    short m_pix_op_inc_w_4
11934                                  <1>
11935                                  <1> m_pix_op_inc_w_1:
11936 0000FA52 803D[71120300]18        <1>         cmp    byte [v_bpp], 24 ; 24bpp
11937 0000FA59 7710                    <1>         ja     short m_pix_op_inc_w_3 ; 32bpp
11938 0000FA5B 7207                    <1>         jb     short m_pix_op_inc_w_2 ; 16bpp
11939                                  <1>
11940                                  <1>         ; 24 bit true colors
11941 0000FA5D BD[D6F90000]            <1>         mov    ebp, m_pix_op_inc_24
11942 0000FA62 EB0C                    <1>         jmp    short m_pix_op_inc_w_4
11943                                  <1>
11944                                  <1>         ; 65536 colors (16bpp)
11945                                  <1> m_pix_op_inc_w_2:
11946 0000FA64 BD[06FA0000]            <1>         mov    ebp, m_pix_op_inc_16
11947 0000FA69 EB05                    <1>         jmp    short m_pix_op_inc_w_4
11948                                  <1>
11949                                  <1>         ; 32 bit true colors
11950                                  <1> m_pix_op_inc_w_3:
11951 0000FA6B BD[26FA0000]            <1>         mov    ebp, m_pix_op_inc_32
11952                                  <1> m_pix_op_inc_w_4:
11953 0000FA70 E9DBF8FFFF              <1>         jmp    m_pix_op_inc_w_x
11954                                  <1>
11955                                  <1> m_pix_op_dec:
11956                                  <1>         ; 06/02/2021
11957                                  <1>         ; DECREASE COLOR (MASKED, full screen)
11958                                  <1>         ;
11959                                  <1>         ; jump from pix_op_dec
11960                                  <1>         ;
11961                                  <1>         ; INPUT:
11962                                  <1>         ;   ecx = [v_siz] ; display page pixel count
11963                                  <1>         ;   esi = edi = [v_mem] ; LFB start address
11964                                  <1>         ;
11965                                  <1>         ;   [maskcolor] = mask color (to be excluded)
11966                                  <1>         ;
11967                                  <1>         ; OUTPUT:
11968                                  <1>         ;     [u.r0] will be > 0 if succesful
11969                                  <1>
11970                                  <1>         ; Full screen
11971                                  <1> m_pix_op_dec_0:
11972 0000FA75 803D[71120300]08        <1>         cmp    byte [v_bpp], 8 ; 8bpp
11973 0000FA7C 7719                    <1>         ja     short m_pix_op_dec_1
```

```
11974                                  <1>       ; 256 colors (8bpp)
11975                                  <1>       ;jmp   short m_pix_op_dec_8
11976                                  <1> m_pix_op_dec_8:
11977                                  <1>       ; 8 bit colors (256 colors)
11978 0000FA7E AC                     <1>       lodsb
11979 0000FA7F 3A05[82120300]          <1>       cmp    al, [maskcolor]
11980 0000FA85 740C                    <1>       je     short m_pix_op_dec_8_1 ; exclude
11981 0000FA87 FE0F                    <1>       dec    byte [edi]
11982 0000FA89 7902                    <1>       jns    short m_pix_op_dec_8_0
11983 0000FA8B FE07                    <1>       inc    byte [edi]
11984                                  <1> m_pix_op_dec_8_0:
11985 0000FA8D FF05[64030300]          <1>       inc    dword [u.r0] ; +1
11986                                  <1> m_pix_op_dec_8_1:
11987 0000FA93 47                      <1>       inc    edi
11988 0000FA94 E2E8                    <1>       loop   m_pix_op_dec_8
11989 0000FA96 C3                      <1>       retn
11990                                  <1> m_pix_op_dec_1:
11991 0000FA97 803D[71120300]18        <1>       cmp    byte [v_bpp], 24 ; 24bpp
11992 0000FA9E 774D                    <1>       ja     short m_pix_op_dec_3 ; 32bpp
11993 0000FAA0 722B                    <1>       jb     short m_pix_op_dec_2 ; 16bpp
11994                                  <1>       ; 24 bit true colors
11995                                  <1>       ;jmp   short m_pix_op_dec_24
11996                                  <1> m_pix_op_dec_24:
11997                                  <1>       ; 24 bit true colors
11998 0000FAA2 66AD                    <1>       lodsw
11999 0000FAA4 C1E010                  <1>       shl    eax, 16
12000 0000FAA7 AC                      <1>       lodsb
12001 0000FAA8 C1C010                  <1>       rol    eax, 16
12002 0000FAAB 3B05[82120300]          <1>       cmp    eax, [maskcolor]
12003 0000FAB1 7414                    <1>       je     short m_pix_op_dec_24_1 ; exclude
12004 0000FAB3 48                      <1>       dec    eax
12005 0000FAB4 7901                    <1>       jns    short m_pix_op_dec_24_0
12006 0000FAB6 40                      <1>       inc    eax
12007                                  <1> m_pix_op_dec_24_0:
12008 0000FAB7 668907                  <1>       mov    [edi], ax
12009 0000FABA C1E810                  <1>       shr    eax, 16
12010 0000FABD 884702                  <1>       mov    [edi+2], al
12011 0000FAC0 8305[64030300]03        <1>       add    dword [u.r0], 3 ; +3
12012                                  <1> m_pix_op_dec_24_1:
12013 0000FAC7 83C703                  <1>       add    edi, 3 ; +3
12014 0000FACA E2D6                    <1>       loop   m_pix_op_dec_24
12015 0000FACC C3                      <1>       retn
12016                                  <1>       ; 65536 colors (16bpp)
12017                                  <1> m_pix_op_dec_2:
12018                                  <1>       ;jmp   short m_pix_op_dec_16
12019                                  <1> m_pix_op_dec_16:
12020                                  <1>       ; 16 bit colors (65536 colors)
12021 0000FACD 66AD                    <1>       lodsw
12022 0000FACF 663B05[82120300]        <1>       cmp    ax, [maskcolor]
12023 0000FAD6 740F                    <1>       je     short m_pix_op_dec_16_1 ; exclude
12024 0000FAD8 66FF0F                  <1>       dec    word [edi]
12025 0000FADB 7903                    <1>       jns    short m_pix_op_dec_16_0
12026 0000FADD 66FF07                  <1>       inc    word [edi]
12027                                  <1> m_pix_op_dec_16_0:
12028 0000FAE0 8305[64030300]02        <1>       add    dword [u.r0], 2 ; +2
12029                                  <1> m_pix_op_dec_16_1:
12030 0000FAE7 83C702                  <1>       add    edi, 2 ; +2
12031 0000FAEA E2E1                    <1>       loop   m_pix_op_dec_16
12032 0000FAEC C3                      <1>       retn
12033                                  <1> m_pix_op_dec_3:
12034                                  <1>       ; 32 bit true colors
12035                                  <1>       ;jmp   short m_pix_op_dec_32
12036                                  <1> m_pix_op_dec_32:
12037                                  <1>       ; 32 bit true colors
12038 0000FAED AD                      <1>       lodsd
12039 0000FAEE 3B05[82120300]          <1>       cmp    eax, [maskcolor]
12040 0000FAF4 740D                    <1>       je     short m_pix_op_dec_32_1 ; exclude
12041 0000FAF6 FF0F                    <1>       dec    dword [edi]
12042 0000FAF8 7902                    <1>       jns    short m_pix_op_dec_32_0
12043 0000FAFA FF07                    <1>       inc    dword [edi]
12044                                  <1> m_pix_op_dec_32_0:
12045 0000FAFC 8305[64030300]04        <1>       add    dword [u.r0], 4 ; +4
12046                                  <1> m_pix_op_dec_32_1:
12047 0000FB03 83C704                  <1>       add    edi, 4 ; +4
12048 0000FB06 E2E5                    <1>       loop   m_pix_op_dec_32
12049 0000FB08 C3                      <1>       retn
12050                                  <1>
12051                                  <1> m_pix_op_dec_w:
12052                                  <1>       ; 06/02/2021
12053                                  <1>       ; DECREASE COLOR (MASKED, window)
12054                                  <1>       ;
12055                                  <1>       ; jump from pix_op_dec_w
12056                                  <1>       ;
12057                                  <1>       ; INPUT:
12058                                  <1>       ;   ecx = bytes per row (to be applied)
12059                                  <1>       ;   edx = screen width in bytes
12060                                  <1>       ;   ebx = row count
12061                                  <1>       ;
12062                                  <1>       ;   [maskcolor] = mask color (to be excluded)
12063                                  <1>       ;
12064                                  <1>       ; OUTPUT:
12065                                  <1>       ;      [u.r0] will be > 0 if succesful
12066                                  <1>
12067                                  <1>       ; window
12068                                  <1>       ;mov   edi, [v_str] ; LFB start address
12069                                  <1>       ;mov   esi, edi
12070                                  <1>
12071 0000FB09 803D[71120300]08        <1>       cmp    byte [v_bpp], 8 ; 8bpp
12072 0000FB10 7707                    <1>       ja     short m_pix_op_dec_w_1
12073                                  <1>
12074                                  <1>       ; 256 colors (8bpp)
12075 0000FB12 BD[7EFA0000]            <1>       mov    ebp, m_pix_op_dec_8
12076 0000FB17 EB1E                    <1>       jmp    short m_pix_op_dec_w_4
12077                                  <1>
12078                                  <1> m_pix_op_dec_w_1:
```

```
12079 0000FB19 803D[71120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
12080 0000FB20 7710                <1>       ja     short m_pix_op_dec_w_3 ; 32bpp
12081 0000FB22 7207                <1>       jb     short m_pix_op_dec_w_2 ; 16bpp
12082                              <1>
12083                              <1>       ; 24 bit true colors
12084 0000FB24 BD[A2FA0000]        <1>       mov    ebp, m_pix_op_dec_24
12085 0000FB29 EB0C                <1>       jmp    short m_pix_op_dec_w_4
12086                              <1>
12087                              <1>       ; 65536 colors (16bpp)
12088                              <1> m_pix_op_dec_w_2:
12089 0000FB2B BD[CDFA0000]        <1>       mov    ebp, m_pix_op_dec_16
12090 0000FB30 EB05                <1>       jmp    short m_pix_op_dec_w_4
12091                              <1>
12092                              <1>       ; 32 bit true colors
12093                              <1> m_pix_op_dec_w_3:
12094 0000FB32 BD[EDFA0000]        <1>       mov    ebp, m_pix_op_dec_32
12095                              <1> m_pix_op_dec_w_4:
12096 0000FB37 E914F8FFFF          <1>       jmp    m_pix_op_dec_w_x
12097                              <1>
12098                              <1> sysvideo_39:
12099                              <1>       ; 15/02/2021
12100                              <1>       ; 07/02/2021, 08/02/2021
12101                              <1>       ; 03/01/2021, 04/01/2021
12102                              <1>       ; 23/11/2020
12103                              <1>       ; BH = 3
12104                              <1>       ; PIXEL READ/WRITE
12105                              <1>
12106                              <1>       ; 07/02/2021
12107                              <1>       ; 04/01/2021 (TRDOS 386 v2.0.3)
12108 0000FB3C 80FB03              <1>       cmp    bl, 3
12109 0000FB3F 761A                <1>       jna    short sysvideo_39_1
12110                              <1>       ; 07/02/2021
12111 0000FB41 80FB06              <1>       cmp    bl, 6
12112 0000FB44 7705                <1>       ja     short sysvideo_39_0
12113 0000FB46 E91A010000          <1>       jmp    sysvideo_39_31
12114                              <1> sysvideo_39_0:
12115                              <1>       ; error
12116 0000FB4B B3FF                <1>       mov    bl, 0FFh
12117 0000FB4D 8B2D[60030300]      <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
12118 0000FB53 895D10              <1>       mov    [ebp+16], ebx ; EBX
12119 0000FB56 E975D8FFFF          <1>       jmp    sysret
12120                              <1> sysvideo_39_1:
12121 0000FB5B 803D[BA6A0000]FF    <1>       cmp    byte [CRT_MODE], 0FFh
12122 0000FB62 7312                <1>       jnb    short sysvideo_39_2 ; SVGA (VESA VBE) video mode
12123                              <1>
12124                              <1>       ; Std VGA or CGA mode
12125 0000FB64 81C200000A00        <1>       add    edx, 0A0000h
12126 0000FB6A 72DF                <1>       jc     short sysvideo_39_0
12127 0000FB6C 81FAFFFF0A00        <1>       cmp    edx, 0AFFFFh
12128 0000FB72 77D7                <1>       ja     short sysvideo_39_0
12129 0000FB74 EB1E                <1>       jmp    short sysvideo_39_3 ; 8bpp
12130                              <1>
12131                              <1> sysvideo_39_2:
12132                              <1>       ; use current vbe (svga) video mode
12133                              <1>
12134                              <1>       ; get LFB address
12135 0000FB76 A1[14120300]        <1>       mov    eax, [LFB_ADDR] ; [LFB_Info+LFBINFO.LFB_addr]
12136 0000FB7B 09C0                <1>       or     eax, eax
12137 0000FB7D 74CC                <1>       jz     short sysvideo_39_0
12138 0000FB7F 3B15[18120300]      <1>       cmp    edx, [LFB_SIZE]
12139 0000FB85 73C4                <1>       jnb    short sysvideo_39_0
12140                              <1>
12141 0000FB87 01C2                <1>       add    edx, eax
12142                              <1>       ;jc    short sysvideo_39_0
12143                              <1>
12144                              <1>       ; Pixel read/write in VESA VBE (2/3) video mode
12145                              <1>       ; Video memory at Linear Frame Buffer base address
12146                              <1>
12147 0000FB89 8A3D[20120300]      <1>       mov    bh, [LFB_Info+LFBINFO.bpp]
12148                              <1>
12149 0000FB8F 80FF08              <1>       cmp    bh, 8 ; 8bpp
12150 0000FB92 775D                <1>       ja     short sysvideo_39_17
12151                              <1>
12152                              <1>       ; 8 bits per pixel
12153                              <1> sysvideo_39_3:
12154 0000FB94 80FB01              <1>       cmp    bl, 1 ; 1 = write pixel
12155 0000FB97 7406                <1>       je     short sysvideo_39_5
12156 0000FB99 7712                <1>       ja     short sysvideo_39_8
12157                              <1> sysvideo_39_4:
12158                              <1>       ; read pixel (8bpp)
12159 0000FB9B 8A02                <1>       mov    al, [edx]
12160                              <1>       ;mov   [u.r0], al
12161                              <1>       ;jmp   sysret
12162 0000FB9D EB04                <1>       jmp    short sysvideo_39_7
12163                              <1> sysvideo_39_5:
12164                              <1>       ; write pixel (8bpp)
12165 0000FB9F 88C8                <1>       mov    al, cl
12166                              <1> sysvideo_39_6:
12167 0000FBA1 8802                <1>       mov    [edx], al
12168                              <1> sysvideo_39_7:
12169 0000FBA3 A2[64030300]        <1>       mov    [u.r0], al
12170 0000FBA8 E923D8FFFF          <1>       jmp    sysret
12171                              <1> sysvideo_39_8:
12172 0000FBAD 80FB03              <1>       cmp    bl, 3 ; mix
12173 0000FBB0 7208                <1>       jb     short sysvideo_39_9
12174                              <1>       ; mix  pixel colors (8bpp)
12175 0000FBB2 8A02                <1>       mov    al, [edx]
12176 0000FBB4 00C8                <1>       add    al, cl
12177 0000FBB6 D0D8                <1>       rcr    al, 1
12178 0000FBB8 EBE7                <1>       jmp    short sysvideo_39_6
12179                              <1> sysvideo_39_9:
12180                              <1>        ; swap pixel colors (8bpp)
12181 0000FBBA 88C8                <1>       mov    al, cl
12182 0000FBBC 8602                <1>       xchg   [edx], al
12183 0000FBBE EBE3                <1>       jmp    short sysvideo_39_7
```

```
12184                                  <1>
12185                                  <1>       ; 16 bits per pixel
12186                                  <1> sysvideo_39_10:
12187 0000FBC0 80FB01                  <1>       cmp    bl, 1 ; 1 = write pixel
12188 0000FBC3 7406                    <1>       je     short sysvideo_39_12
12189 0000FBC5 7714                    <1>       ja     short sysvideo_39_15
12190                                  <1> sysvideo_39_11:
12191                                  <1>       ; read pixel (16bpp)
12192 0000FBC7 8B02                    <1>       mov    eax, [edx]
12193                                  <1>       ;mov   [u.r0], ax
12194                                  <1>       ;jmp   sysret
12195 0000FBC9 EB05                    <1>       jmp    short sysvideo_39_14
12196                                  <1> sysvideo_39_12:
12197                                  <1>       ; write pixel (16bpp)
12198 0000FBCB 89C8                    <1>       mov    eax, ecx
12199                                  <1> sysvideo_39_13:
12200 0000FBCD 668902                  <1>       mov    [edx], ax
12201                                  <1> sysvideo_39_14:
12202 0000FBD0 66A3[64030300]          <1>       mov    [u.r0], ax
12203 0000FBD6 E9F5D7FFFF              <1>       jmp    sysret
12204                                  <1> sysvideo_39_15:
12205 0000FBDB 80FB03                  <1>       cmp    bl, 3 ; mix
12206 0000FBDE 720A                    <1>       jb     short sysvideo_39_16
12207                                  <1>       ; mix  pixel colors (16bpp)
12208 0000FBE0 8B02                    <1>       mov    eax, [edx]
12209 0000FBE2 6601C8                  <1>       add    ax, cx
12210 0000FBE5 66D1D8                  <1>       rcr    ax, 1
12211 0000FBE8 EBE3                    <1>       jmp    short sysvideo_39_13
12212                                  <1> sysvideo_39_16:
12213                                  <1>        ; swap pixel colors (16bpp)
12214 0000FBEA 89C8                    <1>       mov    eax, ecx
12215 0000FBEC 668702                  <1>       xchg   [edx], ax
12216 0000FBEF EBDF                    <1>       jmp    short sysvideo_39_14
12217                                  <1> sysvideo_39_17:
12218 0000FBF1 80FF18                  <1>       cmp    bh, 24
12219 0000FBF4 7743                    <1>       ja     short sysvideo_39_24
12220 0000FBF6 72C8                    <1>       jb     short sysvideo_39_10
12221                                  <1>
12222                                  <1>       ; 24 bits per pixel
12223 0000FBF8 81E1FFFFFF00            <1>       and    ecx, 0FFFFFFh
12224 0000FBFE 80FB01                  <1>       cmp    bl, 1 ; 1 = write pixel
12225 0000FC01 7406                    <1>       je     short sysvideo_39_19
12226 0000FC03 7712                    <1>       ja     short sysvideo_39_22
12227                                  <1> sysvideo_39_18:
12228                                  <1>       ; read pixel (24bpp)
12229 0000FC05 8B02                    <1>       mov    eax, [edx]
12230                                  <1>       ;and   eax, 0FFFFFFh
12231                                  <1>       ;mov   [u.r0], eax
12232                                  <1>       ;jmp   sysret
12233 0000FC07 EB04                    <1>       jmp    short sysvideo_39_21
12234                                  <1> sysvideo_39_19:
12235                                  <1>       ; write pixel (24bpp)
12236 0000FC09 89C8                    <1>       mov    eax, ecx
12237                                  <1> sysvideo_39_20:
12238                                  <1>       ;and   eax, 0FFFFFFh
12239 0000FC0B 8902                    <1>       mov    [edx], eax
12240                                  <1> sysvideo_39_21:
12241 0000FC0D A3[64030300]            <1>       mov    [u.r0], eax
12242 0000FC12 E9B9D7FFFF              <1>       jmp    sysret
12243                                  <1> sysvideo_39_22:
12244 0000FC17 80FB03                  <1>       cmp    bl, 3 ; mix
12245 0000FC1A 720D                    <1>       jb     short sysvideo_39_23
12246                                  <1>       ; mix  pixel colors (24bpp)
12247 0000FC1C 8B02                    <1>       mov    eax, [edx]
12248 0000FC1E 25FFFFFF00              <1>       and    eax, 0FFFFFFh
12249                                  <1>       ;and   ecx, 0FFFFFFh
12250 0000FC23 01C8                    <1>       add    eax, ecx
12251 0000FC25 D1D8                    <1>       rcr    eax, 1
12252 0000FC27 EBE2                    <1>       jmp    short sysvideo_39_20
12253                                  <1> sysvideo_39_23:
12254                                  <1>        ; swap pixel colors (24bpp)
12255 0000FC29 89C8                    <1>       mov    eax, ecx
12256                                  <1>       ;and   eax, 0FFFFFFh
12257 0000FC2B 668702                  <1>       xchg   [edx], ax
12258 0000FC2E C1C810                  <1>       ror    eax, 16
12259 0000FC31 884202                  <1>       mov    [edx+2], al
12260 0000FC34 C1C010                  <1>       rol    eax, 16
12261 0000FC37 EBD4                    <1>       jmp    short sysvideo_39_21
12262                                  <1>
12263                                  <1>       ; 32 bits per pixel
12264                                  <1> sysvideo_39_24:
12265 0000FC39 80FB01                  <1>       cmp    bl, 1 ; 1 = write pixel
12266 0000FC3C 7406                    <1>       je     short sysvideo_39_26
12267 0000FC3E 7712                    <1>       ja     short sysvideo_39_29
12268                                  <1> sysvideo_39_25:
12269                                  <1>       ; read pixel (32bpp)
12270 0000FC40 8B02                    <1>       mov    eax, [edx]
12271                                  <1>       ;mov   [u.r0], eax
12272                                  <1>       ;jmp   sysret
12273 0000FC42 EB04                    <1>       jmp    short sysvideo_39_28
12274                                  <1> sysvideo_39_26:
12275                                  <1>       ; write pixel (32bpp)
12276 0000FC44 89C8                    <1>       mov    eax, ecx
12277                                  <1> sysvideo_39_27:
12278 0000FC46 8902                    <1>       mov    [edx], eax
12279                                  <1> sysvideo_39_28:
12280 0000FC48 A3[64030300]            <1>       mov    [u.r0], eax
12281 0000FC4D E97ED7FFFF              <1>       jmp    sysret
12282                                  <1> sysvideo_39_29:
12283 0000FC52 80FB03                  <1>       cmp    bl, 3 ; mix
12284 0000FC55 7208                    <1>       jb     short sysvideo_39_30
12285                                  <1>       ; mix  pixel colors (32bpp)
12286 0000FC57 8B02                    <1>       mov    eax, [edx]
12287 0000FC59 01C8                    <1>       add    eax, ecx
12288 0000FC5B D1D8                    <1>       rcr    eax, 1
```

```
12289 0000FC5D EBE7               <1>       jmp    short sysvideo_39_27
12290                             <1> sysvideo_39_30:
12291                             <1>        ; swap pixel colors (32bpp)
12292 0000FC5F 89C8               <1>       mov    eax, ecx
12293 0000FC61 8702               <1>       xchg   [edx], eax
12294 0000FC63 EBE3               <1>       jmp    short sysvideo_39_28
12295                             <1>
12296                             <1> sysvideo_39_31:
12297                             <1>       ; 06/03/2021
12298                             <1>       ; 08/02/2021
12299                             <1>       ; 07/02/2021
12300                             <1>       ; BL = 4 -> read pixels from user defined positions
12301                             <1>       ; BL = 5 -> write single color pixels to user defined pos.
12302                             <1>       ; BL = 6 -> write multi color pixels to user defined pos.
12303                             <1>       ; ECX = color (CL, CX, ECX)
12304                             <1>       ; EDX = number of pixels
12305                             <1>       ; ESI = user buffer contains dword pixel positions
12306                             <1>       ;       (and dword colors for BL input = 6)
12307                             <1>       ; EDI = user's pixel color buff (destination) for BL = 4
12308                             <1>
12309 0000FC65 890D[82120300]     <1>       mov    [maskcolor], ecx
12310 0000FC6B 89D5               <1>       mov    ebp, edx ; number of pixels
12311 0000FC6D 803D[BA6A0000]FF   <1>       cmp    byte [CRT_MODE], 0FFh ; SVGA flag
12312 0000FC74 7317               <1>       jnb    short sysvideo_39_33 ; SVGA (VESA VBE mode)
12313                             <1>       ; Standard VGA mode
12314 0000FC76 B900000100         <1>       mov    ecx, 65536 ; Video page size (maximum)
12315 0000FC7B 39CA               <1>       cmp    edx, ecx
12316 0000FC7D 7709               <1>       ja     short sysvideo_39_32 ; abnormal value !
12317 0000FC7F B800000A00         <1>       mov    eax, 0A0000h ; Video page start address
12318 0000FC84 B708               <1>       mov    bh, 8  ; 8 bits per pixel (256 colors)
12319 0000FC86 EB35               <1>       jmp    short sysvideo_39_34
12320                             <1> sysvideo_39_32:
12321                             <1>       ; nonsense! (edx has abnormal value)
12322 0000FC88 E943D7FFFF         <1>       jmp    sysret
12323                             <1> sysvideo_39_33:
12324                             <1>       ; 06/03/2021
12325 0000FC8D 8A3D[20120300]     <1>       mov    bh, [LFB_Info+LFBINFO.bpp]
12326 0000FC93 80FF08             <1>       cmp    bh, 8
12327 0000FC96 7412               <1>       je     short sysvideo_39_81 ; 8bpp
12328 0000FC98 89D0               <1>       mov    eax, edx
12329 0000FC9A 80FF10             <1>       cmp    bh, 16
12330 0000FC9D 7409               <1>       je     short sysvideo_39_80 ; 16bpp
12331 0000FC9F D1E2               <1>       shl    edx, 1
12332 0000FCA1 80FF20             <1>       cmp    bh, 32
12333 0000FCA4 7502               <1>       jne    short sysvideo_39_80 ; 24bpp
12334 0000FCA6 D1E0               <1>       shl    eax, 1
12335                             <1> sysvideo_39_80:
12336 0000FCA8 01C2               <1>       add    edx, eax
12337                             <1>       ; edx = number of bytes
12338                             <1> sysvideo_39_81:
12339                             <1>       ; get LFB address
12340 0000FCAA A1[14120300]       <1>       mov    eax, [LFB_ADDR] ; [LFB_Info+LFBINFO.LFB_addr]
12341 0000FCAF 09C0               <1>       or     eax, eax
12342 0000FCB1 74D5               <1>       jz     short sysvideo_39_32 ; LFB is not ready !
12343 0000FCB3 8B0D[18120300]     <1>       mov    ecx, [LFB_SIZE]
12344 0000FCB9 39CA               <1>       cmp    edx, ecx
12345 0000FCBB 77CB               <1>       ja     short sysvideo_39_32 ; abnormal value !
12346                             <1>
12347                             <1>       ; 02/03/2021
12348                             <1>       ; 08/02/2021
12349                             <1>       ;mov   ebp, edx ; pixel count
12350                             <1>       ;shl   ebp, 2 ; byte count (pixel pos: 4 bytes)
12351                             <1>
12352                             <1>       ; 06/03/2021
12353                             <1>       ; bits per pixel (pixel color size)
12354                             <1>       ;mov   bh, [LFB_Info+LFBINFO.bpp]
12355                             <1> sysvideo_39_34:
12356 0000FCBD C1E502             <1>       shl    ebp, 2 ; 15/02/2021 (byte count)
12357 0000FCC0 A3[72120300]       <1>       mov    [v_mem], eax ; Save video page start address
12358 0000FCC5 88DE               <1>       mov    dh, bl ; sub function
12359                             <1>       ; 06/03/2021
12360 0000FCC7 883D[71120300]     <1>       mov    [v_bpp], bh ; bits per pixel (color size)
12361                             <1>       ;mov   ebx, [LFB_SIZE]
12362 0000FCCD 89CB               <1>       mov    ebx, ecx ; [LFB_SIZE]
12363                             <1>
12364 0000FCCF B900080000         <1>       mov    ecx, 2048
12365 0000FCD4 39CD               <1>       cmp    ebp, ecx
12366 0000FCD6 7302               <1>       jnb    short sysvideo_39_35
12367 0000FCD8 89E9               <1>       mov    ecx, ebp ; fix to requested byte count
12368                             <1> sysvideo_39_35:
12369 0000FCDA 80FE04             <1>       cmp    dh, 4 ; 08/02/2021
12370                             <1>       ;cmp   bl, 4 ; read pixels from user defined positions
12371 0000FCDD 7605               <1>       jna    short sysvideo_39_36
12372 0000FCDF E9B2000000         <1>       jmp    sysvideo_39_52
12373                             <1>       ; 08/02/2021
12374                             <1>       ;mov   [buffer8], edi  ; user's destination buff addr
12375                             <1> sysvideo_39_36:
12376                             <1>       ; 08/02/2021
12377                             <1>       ; read pixel positions
12378                             <1>       ; as defined in user's source buffer
12379 0000FCE4 893D[8A120300]     <1>       mov    [buffer8], edi  ; user's destination buff addr
12380 0000FCEA BF00760900         <1>       mov    edi, VBE3SAVERESTOREBLOCK ; kernel buffer for
12381                             <1>                                     ; 2028 byte data
12382                             <1>       ; esi = user's source buffer for pixel positions
12383                             <1>       ; ecx = byte count
12384 0000FCEF E898180000         <1>       call   transfer_from_user_buffer
12385 0000FCF4 7292               <1>       jc     short sysvideo_39_32 ; error
12386                             <1>       ; ecx  = transfer count (bytes)
12387                             <1>
12388 0000FCF6 57                 <1>       push   edi ; *
12389 0000FCF7 56                 <1>       push   esi ; **
12390 0000FCF8 51                 <1>       push   ecx ; ***
12391                             <1>
12392 0000FCF9 89FE               <1>       mov    esi, edi ; kernel buffer
12393 0000FCFB 8B15[72120300]     <1>       mov    edx, [v_mem] ; video memory
```

```
12394 0000FD01 C1E902              <1>       shr    ecx, 2 ; pixel count (within buffer capacity)
12395                              <1>
12396 0000FD04 803D[71120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
12397 0000FD0B 7753                <1>       ja     short sysvideo_39_49
12398                              <1> sysvideo_39_37:
12399                              <1>       ; 8bpp
12400 0000FD0D AD                  <1>       lodsd
12401 0000FD0E 39D8                <1>       cmp    eax, ebx ; < [LFB_SIZE]
12402 0000FD10 7309                <1>       jnb    short sysvideo_39_39
12403 0000FD12 0FB60402            <1>       movzx  eax, byte [edx+eax]
12404                              <1> sysvideo_39_38:
12405 0000FD16 AB                  <1>       stosd
12406 0000FD17 E2F4                <1>       loop   sysvideo_39_37
12407 0000FD19 EB49                <1>       jmp    short sysvideo_39_50
12408                              <1> sysvideo_39_39:
12409                              <1>       ; write black color for improper positions
12410 0000FD1B 31C0                <1>       xor    eax, eax
12411 0000FD1D EBF7                <1>       jmp    short sysvideo_39_38
12412                              <1> sysvideo_39_40:
12413 0000FD1F 803D[71120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
12414 0000FD26 772A                <1>       ja     short sysvideo_39_47 ; 32bpp
12415 0000FD28 7216                <1>       jb     short sysvideo_39_44 ; 16bpp
12416                              <1> sysvideo_39_41:
12417                              <1>       ; 24bpp
12418 0000FD2A AD                  <1>       lodsd
12419 0000FD2B 39D8                <1>       cmp    eax, ebx ; < [LFB_SIZE]
12420 0000FD2D 730D                <1>       jnb    short sysvideo_39_43
12421 0000FD2F 8B0402              <1>       mov    eax, [edx+eax]
12422 0000FD32 25FFFFFF00          <1>       and    eax, 0FFFFFFh
12423                              <1> sysvideo_39_42:
12424 0000FD37 AB                  <1>       stosd
12425 0000FD38 E2F0                <1>       loop   sysvideo_39_41
12426 0000FD3A EB28                <1>       jmp    short sysvideo_39_50
12427                              <1> sysvideo_39_43:
12428                              <1>       ; write black color for improper positions
12429 0000FD3C 31C0                <1>       xor    eax, eax
12430 0000FD3E EBF7                <1>       jmp    short sysvideo_39_42
12431                              <1> sysvideo_39_44:
12432                              <1>       ; 16bpp
12433 0000FD40 AD                  <1>       lodsd
12434 0000FD41 39D8                <1>       cmp    eax, ebx ; < [LFB_SIZE]
12435 0000FD43 7309                <1>       jnb    short sysvideo_39_46
12436 0000FD45 0FB70402            <1>       movzx  eax, word [edx+eax]
12437                              <1> sysvideo_39_45:
12438 0000FD49 AB                  <1>       stosd
12439 0000FD4A E2F4                <1>       loop   sysvideo_39_44
12440 0000FD4C EB16                <1>       jmp    short sysvideo_39_50
12441                              <1> sysvideo_39_46:
12442                              <1>       ; write black color for improper positions
12443 0000FD4E 31C0                <1>       xor    eax, eax
12444 0000FD50 EBF7                <1>       jmp    short sysvideo_39_45
12445                              <1> sysvideo_39_47:
12446                              <1>       ; 32bpp
12447 0000FD52 AD                  <1>       lodsd
12448 0000FD53 39D8                <1>       cmp    eax, ebx ; < [LFB_SIZE]
12449 0000FD55 7309                <1>       jnb    short sysvideo_39_49
12450 0000FD57 0FB70402            <1>       movzx  eax, word [edx+eax]
12451                              <1> sysvideo_39_48:
12452 0000FD5B AB                  <1>       stosd
12453 0000FD5C E2F4                <1>       loop   sysvideo_39_47
12454 0000FD5E EB04                <1>       jmp    short sysvideo_39_50
12455                              <1> sysvideo_39_49:
12456                              <1>       ; write black color for improper positions
12457 0000FD60 31C0                <1>       xor    eax, eax
12458 0000FD62 EBF7                <1>       jmp    short sysvideo_39_48
12459                              <1> sysvideo_39_50:
12460 0000FD64 59                  <1>       pop    ecx ; transfer count in bytes
12461 0000FD65 5E                  <1>       pop    esi ; ** ; kernel buffer
12462                              <1>       ;mov   esi, VBE3SAVERESTOREBLOCK ; kernel buffer for
12463                              <1>                                       ; 2048 byte data
12464 0000FD66 8B3D[8A120300]      <1>       mov    edi, [buffer8]
12465                              <1>       ; edi = user's destination buffer for pixel colors
12466                              <1>       ; ecx = byte count
12467 0000FD6C E8D1170000          <1>       call   transfer_to_user_buffer
12468 0000FD71 5E                  <1>       pop    esi ; *
12469 0000FD72 7266                <1>       jc     short sysvideo_39_56 ; error
12470                              <1>       ; ecx  = transfer count (bytes)
12471 0000FD74 89C8                <1>       mov    eax, ecx
12472 0000FD76 C1E802              <1>       shr    eax, 2
12473 0000FD79 0105[64030300]      <1>       add    [u.r0], eax ; transfer count (in pixels)
12474                              <1>
12475 0000FD7F 29CD                <1>       sub    ebp, ecx
12476 0000FD81 7657                <1>       jna    short sysvideo_39_56 ; completed/finished
12477 0000FD83 01CE                <1>       add    esi, ecx ; next position in source buffer
12478                              <1>       ;add   [buffer8], ecx ; next pos in destination buff
12479 0000FD85 01CF                <1>       add    edi, ecx
12480 0000FD87 66B90008            <1>       mov    cx, 2048 ; new count, limit: kernel buff size
12481 0000FD8B 39CD                <1>       cmp    ebp, ecx ; remain >= limit ?
12482 0000FD8D 7302                <1>       jnb    short sysvideo_39_51 ; yes
12483 0000FD8F 89E9                <1>       mov    ecx, ebp ; fix byte count to remain bytes
12484                              <1> sysvideo_39_51:
12485 0000FD91 E94EFFFFFF          <1>       jmp    sysvideo_39_36
12486                              <1>
12487                              <1> sysvideo_39_52:
12488 0000FD96 80FE05              <1>       cmp    dh, 5 ; 08/02/2021
12489                              <1>       ;cmp   bl, 5 ; write pixels to user defined positions
12490 0000FD99 7605                <1>       jna    short sysvideo_39_53
12491 0000FD9B E9A1000000          <1>       jmp    sysvideo_39_66
12492                              <1> sysvideo_39_53:
12493                              <1>       ; single color pixel writing
12494 0000FDA0 BF00760900          <1>       mov    edi, VBE3SAVERESTOREBLOCK ; kernel buffer for
12495                              <1>                                        ; 2028 byte data
12496                              <1>       ; esi = user's source buffer for pixel positions
12497                              <1>       ; ecx = byte count
12498 0000FDA5 E8E2170000          <1>       call   transfer_from_user_buffer
```

```
12499 0000FDAA 722E                <1>       jc     short sysvideo_39_56 ; error
12500                              <1>       ; ecx = transfer count (bytes)
12501                              <1>
12502                              <1>       ; write pixels by using (user) defined positions
12503                              <1>       ; ecx = byte count (1,2,3,4 times pixel count)
12504                              <1>       ; edi = system buffer address
12505                              <1>
12506 0000FDAC 56                 <1>       push   esi ; *
12507 0000FDAD 51                  <1>       push   ecx ; **
12508                              <1>
12509 0000FDAE 89FE                <1>       mov    esi, edi
12510 0000FDB0 8B3D[72120300]      <1>       mov    edi, [v_mem]
12511                              <1>
12512                              <1>       ; 08/02/2021
12513 0000FDB6 C1E902              <1>       shr    ecx, 2 ; pixel count
12514 0000FDB9 8B15[82120300]      <1>       mov    edx, [maskcolor]
12515                              <1>       ;mov   ebx, [v_siz]
12516                              <1>
12517 0000FDBF 803D[71120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
12518 0000FDC6 7717                <1>       ja     short sysvideo_39_57
12519                              <1> sysvideo_39_54:
12520                              <1>       ; 8bpp
12521 0000FDC8 AD                  <1>       lodsd
12522 0000FDC9 39D8                <1>       cmp    eax, ebx ; < [v_siz]
12523 0000FDCB 7309                <1>       jnb    short sysvideo_39_55
12524 0000FDCD 881407              <1>       mov    [edi+eax], dl
12525                              <1>       ; 06/03/2021
12526 0000FDD0 FF05[64030300]      <1>       inc    dword [u.r0]
12527                              <1> sysvideo_39_55:
12528 0000FDD6 E2F0                <1>       loop   sysvideo_39_54
12529 0000FDD8 EB50                <1>       jmp    short sysvideo_39_64
12530                              <1> sysvideo_39_56:
12531 0000FDDA E9F1D5FFFF          <1>       jmp    sysret
12532                              <1> sysvideo_39_57:
12533 0000FDDF 803D[71120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
12534 0000FDE6 7732                <1>       ja     short sysvideo_39_62 ; 32bpp
12535 0000FDE8 721D                <1>       jb     short sysvideo_39_60 ; 16bpp
12536                              <1> sysvideo_39_58:
12537                              <1>       ; 24bpp
12538 0000FDEA AD                  <1>       lodsd
12539 0000FDEB 39D8                <1>       cmp    eax, ebx ; < [v_siz]
12540 0000FDED 7314                <1>       jnb    short sysvideo_39_59
12541 0000FDEF 881407              <1>       mov    [edi+eax], dl
12542 0000FDF2 40                  <1>       inc    eax
12543 0000FDF3 C1CA08              <1>       ror    edx, 8
12544 0000FDF6 66891407            <1>       mov    [edi+eax], dx
12545 0000FDFA C1C208              <1>       rol    edx, 8
12546 0000FDFD FF05[64030300]      <1>       inc    dword [u.r0]
12547                              <1> sysvideo_39_59:
12548 0000FE03 E2E5                <1>       loop   sysvideo_39_58
12549 0000FE05 EB23                <1>       jmp    short sysvideo_39_64
12550                              <1> sysvideo_39_60:
12551                              <1>       ; 16bpp
12552 0000FE07 AD                  <1>       lodsd
12553 0000FE08 39D8                <1>       cmp    eax, ebx ; < [v_siz]
12554 0000FE0A 730A                <1>       jnb    short sysvideo_39_61
12555 0000FE0C 66891407            <1>       mov    [edi+eax], dx
12556 0000FE10 FF05[64030300]      <1>       inc    dword [u.r0]
12557                              <1> sysvideo_39_61:
12558 0000FE16 E2EF                <1>       loop   sysvideo_39_60
12559 0000FE18 EB10                <1>       jmp    short sysvideo_39_64
12560                              <1> sysvideo_39_62:
12561                              <1>       ; 32bpp
12562 0000FE1A AD                  <1>       lodsd
12563 0000FE1B 39D8                <1>       cmp    eax, ebx ; < [v_siz]
12564 0000FE1D 7309                <1>       jnb    short sysvideo_39_63
12565 0000FE1F 891407              <1>       mov    [edi+eax], edx
12566 0000FE22 FF05[64030300]      <1>       inc    dword [u.r0]
12567                              <1> sysvideo_39_63:
12568 0000FE28 E2F0                <1>       loop   sysvideo_39_62
12569                              <1> sysvideo_39_64:
12570 0000FE2A 59                  <1>       pop    ecx ; **
12571 0000FE2B 5E                  <1>       pop    esi ; *
12572 0000FE2C 29CD                <1>       sub    ebp, ecx
12573 0000FE2E 76AA                <1>       jna    short sysvideo_39_56
12574 0000FE30 01CE                <1>       add    esi, ecx
12575 0000FE32 66B90008            <1>       mov    cx, 2048
12576 0000FE36 39CD                <1>       cmp    ebp, ecx
12577 0000FE38 7302                <1>       jnb    short sysvideo_39_65
12578 0000FE3A 89E9                <1>       mov    ecx, ebp
12579                              <1> sysvideo_39_65:
12580 0000FE3C E95FFFFFFF          <1>       jmp    sysvideo_39_53
12581                              <1>
12582                              <1> sysvideo_39_66:
12583                              <1>       ; 15/02/2021
12584 0000FE41 D1E5                <1>       shl    ebp, 1 ; 8 bytes per pixel (position&color)
12585                              <1> sysvideo_39_67:
12586 0000FE43 66B90008            <1>       mov    cx, 2048
12587 0000FE47 39CD                <1>       cmp    ebp, ecx
12588 0000FE49 7302                <1>       jnb    short sysvideo_39_68
12589 0000FE4B 89E9                <1>       mov    ecx, ebp
12590                              <1> sysvideo_39_68:
12591                              <1>       ; multi colors pixel writing
12592 0000FE4D BF00760900          <1>       mov    edi, VBE3SAVERESTOREBLOCK ; kernel buffer for
12593                              <1>                                     ; 2048 byte data
12594                              <1>       ; esi = user's source buffer for pixel positions
12595                              <1>       ; ecx = byte count
12596 0000FE52 E835170000          <1>       call   transfer_from_user_buffer
12597 0000FE57 7281                <1>       jc     short sysvideo_39_56 ; error
12598                              <1>       ; ecx  = transfer count
12599                              <1>
12600                              <1>       ; write pixels & colors as defined in user buffer
12601                              <1>       ; ecx = byte count (2,4,6,8 times pixel count)
12602                              <1>       ; edi = system buffer address
12603                              <1>
```

```
12604 0000FE59 56                  <1>       push   esi ; **
12605 0000FE5A 51                  <1>       push   ecx ; *
12606                              <1>
12607 0000FE5B 89FE                <1>       mov    esi, edi
12608 0000FE5D 8B3D[72120300]      <1>       mov    edi, [v_mem]
12609                              <1>
12610                              <1>       ; 08/02/2021
12611 0000FE63 C1E903              <1>       shr    ecx, 3 ; pixel count
12612                              <1>
12613                              <1>       ;mov   ebx, [v_siz]
12614                              <1>
12615 0000FE66 803D[71120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
12616 0000FE6D 7715                <1>       ja     short sysvideo_39_71
12617                              <1> sysvideo_39_69:
12618                              <1>       ; 8bpp
12619 0000FE6F AD                  <1>       lodsd  ; position
12620 0000FE70 89C2                <1>       mov    edx, eax
12621 0000FE72 AD                  <1>       lodsd  ; color
12622 0000FE73 39DA                <1>       cmp    edx, ebx ; < [v_siz]
12623 0000FE75 7309                <1>       jnb    short sysvideo_39_70
12624 0000FE77 880417              <1>       mov    [edi+edx], al
12625                              <1>       ; 06/03/2021
12626 0000FE7A FF05[64030300]      <1>       inc    dword [u.r0]
12627                              <1> sysvideo_39_70:
12628 0000FE80 E2ED                <1>       loop   sysvideo_39_69
12629 0000FE82 EB51                <1>       jmp    short sysvideo_39_78
12630                              <1> sysvideo_39_71:
12631 0000FE84 803D[71120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
12632 0000FE8B 7735                <1>       ja     short sysvideo_39_76 ; 32bpp
12633 0000FE8D 721D                <1>       jb     short sysvideo_39_74 ; 16bpp
12634                              <1> sysvideo_39_72:
12635                              <1>       ; 24bpp
12636 0000FE8F AD                  <1>       lodsd  ; position
12637 0000FE90 89C2                <1>       mov    edx, eax
12638 0000FE92 AD                  <1>       lodsd  ; color
12639 0000FE93 39DA                <1>       cmp    edx, ebx ; < [v_siz]
12640 0000FE95 7311                <1>       jnb    short sysvideo_39_73
12641 0000FE97 880417              <1>       mov    [edi+edx], al
12642 0000FE9A 42                  <1>       inc    edx
12643 0000FE9B C1E808              <1>       shr    eax, 8
12644 0000FE9E 66890417            <1>       mov    [edi+edx], ax
12645 0000FEA2 FF05[64030300]      <1>       inc    dword [u.r0]
12646                              <1> sysvideo_39_73:
12647 0000FEA8 E2E5                <1>       loop   sysvideo_39_72
12648 0000FEAA EB29                <1>       jmp    short sysvideo_39_78
12649                              <1> sysvideo_39_74:
12650                              <1>       ; 16bpp
12651 0000FEAC AD                  <1>       lodsd  ; position
12652 0000FEAD 89C2                <1>       mov    edx, eax
12653 0000FEAF AD                  <1>       lodsd  ; color
12654 0000FEB0 39DA                <1>       cmp    edx, ebx ; < [v_siz]
12655 0000FEB2 730A                <1>       jnb    short sysvideo_39_75
12656 0000FEB4 66890417            <1>       mov    [edi+edx], ax
12657 0000FEB8 FF05[64030300]      <1>       inc    dword [u.r0]
12658                              <1> sysvideo_39_75:
12659 0000FEBE E2EC                <1>       loop   sysvideo_39_74
12660 0000FEC0 EB13                <1>       jmp    short sysvideo_39_78
12661                              <1> sysvideo_39_76:
12662                              <1>       ; 32bpp
12663 0000FEC2 AD                  <1>       lodsd  ; position
12664 0000FEC3 89C2                <1>       mov    edx, eax
12665 0000FEC5 AD                  <1>       lodsd  ; color
12666 0000FEC6 39DA                <1>       cmp    edx, ebx ; < [v_siz]
12667 0000FEC8 7309                <1>       jnb    short sysvideo_39_77
12668 0000FECA 890417              <1>       mov    [edi+edx], eax
12669 0000FECD FF05[64030300]      <1>       inc    dword [u.r0]
12670                              <1> sysvideo_39_77:
12671 0000FED3 E2ED                <1>       loop   sysvideo_39_76
12672                              <1> sysvideo_39_78:
12673 0000FED5 59                  <1>       pop    ecx ; *
12674 0000FED6 5E                  <1>       pop    esi ; **
12675                              <1>
12676 0000FED7 29CD                <1>       sub    ebp, ecx
12677 0000FED9 762A                <1>       jna    short sysvideo_39_79
12678 0000FEDB 01CE                <1>       add    esi, ecx
12679 0000FEDD E961FFFFFF          <1>       jmp    sysvideo_39_67
12680                              <1> ;sysvideo_39_79:
12681                              <1> ;     jmp    sysret
12682                              <1>
12683                              <1> sysvideo_16:
12684                              <1>       ; 06/03/2021
12685                              <1>       ; 23/11/2020
12686 0000FEE2 80FF04              <1>       cmp    bh, 4
12687 0000FEE5 0F8251FCFFFF        <1>       jb     sysvideo_39 ; bh = 3, pixel r/w
12688 0000FEEB 771D                <1>       ja     short sysvideo_17
12689                              <1>
12690                              <1>       ; BH = 4
12691                              <1>       ; Direct User Access for CGA video memory.
12692                              <1>       ; Setup user's page tables for direct access to 0B8000h.
12693                              <1>       ;
12694                              <1>       ; Permission checks are not implemented yet !
12695                              <1>       ; (11/07/2016)
12696                              <1>
12697 0000FEED B800800B00          <1>       mov    eax, 0B8000h
12698 0000FEF2 B908000000          <1>       mov    ecx, 8 ; 8 pages (8*4K=32K)
12699 0000FEF7 89C3                <1>       mov    ebx, eax ; 12/05/2017 ; virtual = physical
12700 0000FEF9 E81663FFFF          <1>       call   direct_memory_access
12701                              <1>       ;jc    sysret
12702 0000FEFE 7205                <1>       jc     short sysvideo_39_79 ; 06/03/2021
12703                              <1>       ; eax = 0B8000h if there is not an error
12704 0000FF00 A3[64030300]        <1>       mov    [u.r0], eax
12705                              <1> sysvideo_39_79: ; 08/01/2021
12706 0000FF05 E9C6D4FFFF          <1>       jmp    sysret
12707                              <1>
12708                              <1> sysvideo_17:
```

```
12709                                  <1>       ; 23/12/2020
12710                                  <1>       ; 11/12/2020
12711                                  <1>       ; 10/12/2020
12712                                  <1>       ; 23/11/2020
12713 0000FF0A 80FF06                  <1>       cmp    bh, 6
12714 0000FF0D 740B                    <1>       je     short sysvideo_17_0
12715 0000FF0F 0F82B9000000            <1>       jb     sysvideo_18
12716 0000FF15 E913010000              <1>       jmp    sysvideo_20 ; ja
12717                                  <1>
12718                                  <1> sysvideo_17_0:
12719                                  <1>       ; BH = 6
12720                                  <1>       ; Direct User Access to Linear Frame Buffer.
12721                                  <1>       ; Setup user's page tables for direct access to LFB.
12722                                  <1>       ;
12723                                  <1>       ; Permission checks are not implemented yet !
12724                                  <1>       ; (10/12/2020)
12725                                  <1>
12726 0000FF1A 80FBFF                  <1>       cmp    bl, 0FFh ; current video mode
12727 0000FF1D 722C                    <1>       jb     short sysvideo_17_2 ; for desired video mode
12728                                  <1>
12729 0000FF1F 381D[BA6A0000]          <1>       cmp    [CRT_MODE], bl ; VESA VBE video mode ?
12730 0000FF25 750E                    <1>       jne    short sysvideo_17_1
12731 0000FF27 668B0D[06120300]        <1>       mov    cx, [video_mode]
12732 0000FF2E 6681E1FF01              <1>       and    cx, 1FFh
12733 0000FF33 EB29                    <1>       jmp    short sysvideo_17_3
12734                                  <1> sysvideo_17_1:
12735                                  <1>       ; 11/12/2020
12736 0000FF35 88DF                    <1>       mov    bh, bl ; 0FFh
12737 0000FF37 8A1D[BA6A0000]          <1>       mov    bl, [CRT_MODE] ; VGA/CGA video mode
12738 0000FF3D 8B2D[60030300]          <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
12739                                  <1>       ; 23/12/2020
12740 0000FF43 895D10                  <1>       mov    [ebp+16], ebx ; return to user with EBX value
12741 0000FF46 E985D4FFFF              <1>       jmp    sysret ; return to user with EAX = 0
12742                                  <1> sysvideo_17_2:
12743                                  <1>       ; bl = VESA video mode - 100h
12744 0000FF4B B701                    <1>       mov    bh, 1 ; bx = 1XXh
12745 0000FF4D 53                      <1>       push   ebx ; requested vesa video mode
12746 0000FF4E E89A3BFFFF              <1>       call   vbe_biosfn_return_current_mode
12747 0000FF53 59                      <1>       pop    ecx ; requested vesa video mode
12748 0000FF54 6681E3FF01              <1>       and    bx, 1FFh
12749 0000FF59 6639D9                  <1>       cmp    cx, bx
12750 0000FF5C 7564                    <1>       jne    short sysvideo_17_8
12751                                  <1> sysvideo_17_3:
12752 0000FF5E 663B0D[12120300]        <1>       cmp    cx, [LFB_Info+LFBINFO.mode]
12753 0000FF65 755B                    <1>       jne    short sysvideo_17_8
12754                                  <1> sysvideo_17_4:
12755                                  <1>       ; 11/12/2020
12756 0000FF67 A1[14120300]            <1>       mov    eax, [LFB_Info+LFBINFO.LFB_addr]
12757                                  <1>       ; 21/12/2020
12758 0000FF6C 09C0                    <1>       or     eax, eax
12759 0000FF6E 744D                    <1>       jz     short sysvideo_17_7
12760                                  <1>       ;
12761 0000FF70 8B0D[18120300]          <1>       mov    ecx, [LFB_Info+LFBINFO.LFB_size] ; buff size
12762 0000FF76 89C3                    <1>       mov    ebx, eax ; user's address = physical address
12763                                  <1>       ;push  ebx
12764 0000FF78 51                      <1>       push   ecx
12765                                  <1>       ; 21/12/2020
12766 0000FF79 81C1FF0F0000            <1>       add    ecx, 4095  ; PAGESIZE - 1
12767                                  <1>       ; 14/12/2020
12768 0000FF7F C1E90C                  <1>       shr    ecx, 12  ; convert bytes to pages
12769 0000FF82 E88D62FFFF              <1>       call   direct_memory_access
12770 0000FF87 5A                      <1>       pop    edx  ; linear frame buffer size in bytes
12771                                  <1>       ;pop   eax  ; linear frame buffer address (physical)
12772 0000FF88 7233                    <1>       jc     short sysvideo_17_7 ; [u.r0] = eax = 0
12773                                  <1> sysvideo_17_5:
12774 0000FF8A 668B0D[1E120300]        <1>       mov    cx, [LFB_Info+LFBINFO.Y_res] ; screen height
12775 0000FF91 C1E110                  <1>       shl    ecx, 16
12776 0000FF94 668B0D[1C120300]        <1>       mov    cx,  [LFB_Info+LFBINFO.X_res] ; screen width
12777 0000FF9B 31DB                    <1>       xor    ebx, ebx
12778 0000FF9D 8A1D[20120300]          <1>       mov    bl, [LFB_Info+LFBINFO.bpp] ; bits per pixel
12779 0000FFA3 8A3D[12120300]          <1>       mov    bh, [LFB_Info+LFBINFO.mode] ; XX part of 1XXh
12780                                  <1> sysvideo_26_4: ; 23/12/2020
12781 0000FFA9 8B2D[60030300]          <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
12782 0000FFAF 895514                  <1>       mov    [ebp+20], edx ; return to user with EDX value
12783 0000FFB2 895D10                  <1>       mov    [ebp+16], ebx ; EBX
12784 0000FFB5 894D18                  <1>       mov    [ebp+24], ecx ; ECX
12785                                  <1> sysvideo_17_6:
12786 0000FFB8 A3[64030300]            <1>       mov    [u.r0], eax ; LFB address
12787                                  <1> sysvideo_17_7:
12788 0000FFBD E90ED4FFFF              <1>       jmp    sysret
12789                                  <1> sysvideo_17_8:
12790                                  <1>       ; cx = mode
12791                                  <1>       ; 21/12/2020
12792 0000FFC2 80CD40                  <1>       or     ch, 40h  ; Linear frame buffer flag
12793 0000FFC5 E84539FFFF              <1>       call   _vbe_biosfn_return_mode_info
12794 0000FFCA 72F1                    <1>       jc     short sysvideo_17_7
12795 0000FFCC EB99                    <1>       jmp    short sysvideo_17_4
12796                                  <1>
12797                                  <1> sysvideo_18:
12798                                  <1>       ; BH = 5
12799                                  <1>       ; Direct User Access for VGA video memory.
12800                                  <1>       ; Setup user's page tables for direct access to 0A0000h.
12801                                  <1>       ;
12802                                  <1>       ; Permission checks are not implemented yet !
12803                                  <1>       ; (11/07/2016)
12804                                  <1>
12805 0000FFCE B800000A00              <1>       mov    eax, 0A0000h
12806 0000FFD3 B910000000              <1>       mov    ecx, 16 ; 16 pages (16*4K=64K)
12807 0000FFD8 89C3                    <1>       mov    ebx, eax ; 12/05/2017 ; virtual = physical
12808 0000FFDA E83562FFFF              <1>       call   direct_memory_access
12809 0000FFDF 0F82EBD3FFFF            <1>       jc     sysret
12810                                  <1>       ; eax = 0A0000h if there is not an error
12811 0000FFE5 A3[64030300]            <1>       mov    [u.r0], eax
12812 0000FFEA E9E1D3FFFF              <1>       jmp    sysret
12813                                  <1>
```

```
12814                                  <1> sysvideo_19:
12815                                  <1>       ; 22/01/2021
12816                                  <1>       ; 12/12/2020
12817                                  <1>       ; 11/12/2020
12818                                  <1>       ; 23/11/2020
12819                                  <1>       ; BH = 7
12820                                  <1>       ; Get (Super/Extended VGA) mode
12821                                  <1>       ; and Linear Frame Buffer info.
12822                                  <1>
12823                                  <1>       ; 22/01/2021
12824 0000FFEF B3FF                   <1>       mov    bl, 0FFh
12825                                  <1>       ; 11/12/2020
12826                                  <1>       ;cmp   byte [CRT_MODE], 0FFh ; (extended mode?)
12827                                  <1>       ; 22/01/2021
12828 0000FFF1 381D[BA6A0000]         <1>       cmp    [CRT_MODE], bl  ; 0FFh
12829                                  <1>       ;jb    sysvideo_17_1; not a VESA VBE mode
12830                                  <1>       ; 12/12/2020
12831 0000FFF7 7305                   <1>       jnb    short sysvideo_19_0
12832 0000FFF9 E937FFFFFF             <1>       jmp    sysvideo_17_1
12833                                  <1>
12834                                  <1> sysvideo_19_0:
12835 0000FFFE E8EA3AFFFF             <1>       call   vbe_biosfn_return_current_mode
12836 00010003 6681E3FF01             <1>       and    bx, 1FFh
12837 00010008 663B1D[12120300]       <1>       cmp    bx, [LFB_Info+LFBINFO.mode]
12838 0001000F 7510                   <1>       jne    short sysvideo_19_2
12839                                  <1> sysvideo_19_1:
12840 00010011 A1[14120300]           <1>       mov    eax, [LFB_Info+LFBINFO.LFB_addr]
12841 00010016 8B15[18120300]         <1>       mov    edx, [LFB_Info+LFBINFO.LFB_size]
12842 0001001C E969FFFFFF             <1>       jmp    sysvideo_17_5
12843                                  <1> sysvideo_19_2:
12844 00010021 E8E938FFFF             <1>       call   _vbe_biosfn_return_mode_info
12845 00010026 73E9                   <1>       jnc    short sysvideo_19_1
12846 00010028 E9A3D3FFFF             <1>       jmp    sysret
12847                                  <1>
12848                                  <1> sysvideo_20:
12849                                  <1>       ; 11/12/2020
12850                                  <1>       ; 23/11/2020
12851 0001002D 80FF08                 <1>       cmp    bh, 8
12852 00010030 72BD                   <1>       jb     short sysvideo_19 ; video mode & lfb info
12853 00010032 0F8780000000           <1>       ja     sysvideo_21 ; 12/12/2020
12854                                  <1>
12855                                  <1>       ; BH = 8
12856                                  <1>       ; Set (Super/Extended VGA) mode & return LFB info
12857                                  <1>       ;
12858                                  <1>
12859                                  <1>       ; 11/12/2020
12860 00010038 80FBFF                 <1>       cmp    bl, 0FFh  ; CGA/VGA mode ?
12861 0001003B 7318                   <1>       jnb    short sysvideo_20_1
12862                                  <1>
12863                                  <1>       ;xor   ah, ah
12864 0001003D 88D8                   <1>       mov    al, bl
12865                                  <1> sysvideo_20_0:
12866 0001003F E8B716FFFF             <1>       call   _int10h  ; uses vbe3 pmi32 option
12867 00010044 83F8FF                 <1>       cmp    eax, 0FFFFFFFFh ; -1
12868 00010047 7459                   <1>       je     short sysvideo_20_3 ; error
12869                                  <1>
12870                                  <1>       ; 11/12/2020
12871                                  <1>       ; alternative (it does not use vbe3 pmi32)
12872                                  <1>       ;push  eax
12873                                  <1>       ;call  _set_mode
12874                                  <1>       ;pop   eax
12875                                  <1>       ;jc    short sysvideo_20_3
12876                                  <1>
12877                                  <1>       ;inc   eax
12878 00010049 FEC0                   <1>       inc    al
12879                                  <1>       ;mov   [u.r0], ax ; video mode + 1
12880 0001004B A2[64030300]           <1>       mov    [u.r0], al
12881 00010050 E97BD3FFFF             <1>       jmp    sysret
12882                                  <1>
12883                                  <1> sysvideo_20_1:
12884                                  <1>       ; cx = vesa video mode
12885 00010055 6689C8                 <1>       mov    ax, cx
12886 00010058 663D0001               <1>       cmp    ax, 100h
12887 0001005C 72E1                   <1>       jb     short sysvideo_20_0 ; VGA/CGA mode
12888 0001005E 663DFF01               <1>       cmp    ax, 1FFh
12889                                  <1>       ;ja    short sysvideo_20_4 ; not valid
12890 00010062 773E                   <1>       ja     short sysvideo_20_3
12891 00010064 50                     <1>       push   eax
12892 00010065 6689C3                 <1>       mov    bx, ax
12893 00010068 66B8024F               <1>       mov    ax, 4F02h
12894                                  <1>
12895                                  <1>       ; simulate _int10h (int 31h) for func 4F02h
12896                                  <1>       ;pushfd
12897                                  <1>       ;push  cs
12898                                  <1>       ;push  sysvideo_20_1_retn
12899                                  <1>       ;push  es ; *
12900                                  <1>       ;push  ds ; **      ; SAVE WORK AND PARAMETER REGISTERS
12901                                  <1>       ;jmp   VBE_func
12902                                  <1> ;sysvideo_20_1_retn:
12903                                  <1>
12904 0001006C E88A16FFFF             <1>       call   _int10h ; simulate int 10h (int 31h)
12905                                  <1>
12906 00010071 6683F84F               <1>       cmp    ax, 004Fh
12907 00010075 58                     <1>       pop    eax
12908 00010076 752A                   <1>       jne    short sysvideo_20_3 ; error
12909                                  <1>       ;pop   eax
12910 00010078 40                     <1>       inc    eax
12911 00010079 A3[64030300]           <1>       mov    [u.r0], eax ; video mode + 1
12912 0001007E 09D2                   <1>       or     edx, edx ; is LFBINFO requested by user ?
12913                                  <1>       ;jz    short sysvideo_20_4
12914 00010080 7420                   <1>       jz     short sysvideo_20_3 ; no
12915                                  <1>
12916                                  <1>       ; 11/12/2020
12917                                  <1>       ; Check LFBINFO table/structure
12918                                  <1>       ; (it is set by vbe2 'vbe_biosfn_set_mode'
```

```
12919                                  <1>       ; but if vbe3 vbios pmi is in use,
12920                                  <1>       ; it will not set LFBINFO table)
12921                                  <1>
12922 00010082 52                      <1>       push   edx
12923 00010083 48                      <1>       dec    eax  ; video mode
12924 00010084 BE[12120300]            <1>       mov    esi, LFB_Info
12925 00010089 663B06                  <1>       cmp    ax, [esi+LFBINFO.mode]
12926 0001008C 7407                    <1>       je     short sysvideo_20_2
12927                                  <1>
12928 0001008E E87C38FFFF              <1>       call   _vbe_biosfn_return_mode_info
12929                                  <1>       ;jnc   short sysvideo_20_2
12930 00010093 7212                    <1>       jc     short sysvideo_20_4 ; edx = 0
12931                                  <1>
12932                                  <1>       ;; clear LFBINFO table for invalidating
12933                                  <1>       ;mov   ecx, LFBINFO.size ; 16
12934                                  <1>       ;mov   edi, esi ; LFB_Info table address
12935                                  <1>       ;xor   eax, eax
12936                                  <1>       ;rep   stosb
12937                                  <1>
12938                                  <1> sysvideo_20_2:
12939                                  <1>       ;pop   ecx
12940                                  <1>       ;mov   edi, ecx ; user buffer
12941 00010095 5F                      <1>       pop    edi
12942 00010096 B910000000              <1>       mov    ecx, LFBINFO.size ; 16
12943 0001009B E8A2140000              <1>       call   transfer_to_user_buffer ; fast transfer
12944 000100A0 7206                    <1>       jc     short sysvideo_20_5
12945                                  <1>
12946                                  <1>       ;jmp   sysret
12947                                  <1> sysvideo_20_3:
12948                                  <1>       ;pop   eax ; [u.r0] = 0
12949                                  <1> ;sysvideo_20_4:
12950 000100A2 E929D3FFFF              <1>       jmp    sysret
12951                                  <1>
12952                                  <1> sysvideo_20_4:
12953 000100A7 5A                      <1>       pop    edx
12954                                  <1> sysvideo_20_5:
12955 000100A8 31D2                    <1>       xor    edx, edx ; 0
12956                                  <1>       ; edx = 0 -> invalid LFBINFO data
12957 000100AA 8B2D[60030300]          <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
12958 000100B0 895514                  <1>       mov    [ebp+20], edx ; return to user with EDX value
12959 000100B3 E918D3FFFF              <1>       jmp    sysret
12960                                  <1>
12961                                  <1> sysvideo_21:
12962                                  <1>       ; 04/01/2021
12963                                  <1>       ; 03/12/2020
12964 000100B8 80FF0A                  <1>       cmp    bh, 10
12965 000100BB 0F82AA010000            <1>       jb     sysvideo_22 ; VESA VBE3 pmi parms
12966                                  <1>       ; 23/12/2020
12967 000100C1 0F845F020000            <1>       je     sysvideo_26 ; Video memory mapping
12968                                  <1>
12969                                  <1>       ; 04/01/2020
12970 000100C7 80FF0B                  <1>       cmp    bh, 11
12971 000100CA 0F87E1020000            <1>       ja     sysvideo_27
12972                                  <1>
12973                                  <1>       ; BH = 11
12974                                  <1>       ; set/read DAC color registers (for 8bpp)
12975                                  <1>
12976 000100D0 80FB04                  <1>       cmp    bl, 4
12977 000100D3 0F83AB000000            <1>       jnb    sysvideo_21_7; BMP file type palette
12978                                  <1>                           ; handling
12979 000100D9 F6C301                  <1>       test   bl, 1
12980 000100DC 7555                    <1>       jnz    short sysvideo_21_4 ; set/write DAC colors
12981                                  <1>
12982                                  <1>       ; Read DAC color register or all DAC color registers
12983 000100DE F6C302                  <1>       test   bl, 2  ; read single DAC color register
12984 000100E1 7424                    <1>       jz     short sysvideo_21_2 ; read all DAC color regs
12985                                  <1>
12986                                  <1>       ; read single DAC color register
12987                                  <1>       ; CL = DAC color register (index)
12988                                  <1>
12989 000100E3 66BAC703                <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
12990 000100E7 88C8                    <1>       mov    al, cl ; DAC color register
12991 000100E9 31C9                    <1>       xor    ecx, ecx ; (this may not be necessary)
12992 000100EB EE                      <1>       out    dx, al
12993                                  <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
12994 000100EC B2C9                    <1>       mov    dl, 0C9h
12995 000100EE EC                      <1>       in     al, dx
12996 000100EF 88C4                    <1>       mov    ah, al ; red
12997 000100F1 EC                      <1>       in     al, dx
12998 000100F2 88C1                    <1>       mov    cl, al ; green
12999 000100F4 EC                      <1>       in     al, dx
13000 000100F5 88C5                    <1>       mov    ch, al ; blue
13001 000100F7 C1E108                  <1>       shl    ecx, 8
13002 000100FA 88E1                    <1>       mov    cl, ah ; red
13003                                  <1>       ; CL = Red, CH = Green, byte 3 = Blue, byte 4 = 0
13004                                  <1> sysvideo_21_0:
13005 000100FC 890D[64030300]          <1>       mov    [u.r0], ecx
13006                                  <1> sysvideo_21_1:
13007 00010102 E9C9D2FFFF              <1>       jmp    sysret
13008                                  <1> sysvideo_21_2:
13009                                  <1>       ; read all DAC color registers
13010 00010107 89CB                    <1>       mov    ebx, ecx ; user's buffer address
13011 00010109 BF00600900              <1>       mov    edi, VBE3STACKADDR
13012 0001010E 89FE                    <1>       mov    esi, edi
13013 00010110 B900030000              <1>       mov    ecx, 768 ; 256*3
13014 00010115 51                      <1>       push   ecx
13015 00010116 66BAC703                <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
13016 0001011A 28C0                    <1>       sub    al, al ; 0
13017 0001011C EE                      <1>       out    dx, al
13018                                  <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
13019 0001011D B2C9                    <1>       mov    dl, 0C9h
13020                                  <1> sysvideo_21_3:
13021 0001011F EC                      <1>       in     al, dx
13022 00010120 AA                      <1>       stosb
13023 00010121 EC                      <1>       in     al, dx
```

```
13024 00010122 AA                <1>       stosb
13025 00010123 EC                <1>       in     al, dx
13026 00010124 AA                <1>       stosb
13027 00010125 E2F8              <1>       loop   sysvideo_21_3
13028 00010127 59                <1>       pop    ecx
13029                            <1>
13030 00010128 89DF              <1>       mov    edi, ebx ; user's buffer address
13031                            <1>       ;mov   esi, VBE3STACKADDR
13032                            <1>       ;mov   ecx, 256*3 = 768
13033 0001012A E813140000        <1>       call   transfer_to_user_buffer
13034 0001012F 72D1              <1>       jc     short sysvideo_21_1
13035                            <1>       ;mov   [u.r0], ecx ; actual transfer count
13036 00010131 EBC9              <1>       jmp    short sysvideo_21_0
13037                            <1>
13038                            <1> sysvideo_21_4:
13039                            <1>       ; Set/Write DAC color register or all registers
13040 00010133 F6C302            <1>       test   bl, 2 ; write/set single DAC color register
13041 00010136 741C              <1>       jz     short sysvideo_21_5 ; set all DAC color regs
13042                            <1>
13043                            <1>       ; set single DAC color register
13044                            <1>       ; CL = DAC color register (index)
13045                            <1>       ; (byte 1 = Red, byte 2 = Green, byte 3 = Blue)
13046                            <1>
13047 00010138 66BAC803          <1>       mov    dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
13048 0001013C 89C8              <1>       mov    eax, ecx ; DAC color register (index)
13049 0001013E C1E910            <1>       shr    ecx, 16  ; CL = green, AH = Red
13050 00010141 EE                <1>       out    dx, al
13051                            <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
13052 00010142 FEC2              <1>       inc    dl
13053 00010144 88E0              <1>       mov    al, ah ; Red
13054 00010146 EE                <1>       out    dx, al
13055 00010147 88C8              <1>       mov    al, cl ; Green
13056 00010149 EE                <1>       out    dx, al
13057 0001014A 88E8              <1>       mov    al, ch ; Blue
13058 0001014C EE                <1>       out    dx, al
13059                            <1>       ;rol   ecx, 8
13060 0001014D C1E108            <1>       shl    ecx, 8 ; 21/02/2021
13061 00010150 88E1              <1>       mov    cl, ah ; Red
13062                            <1>              ; ecx = 00BBGGRRh
13063 00010152 EBA8              <1>       jmp    short sysvideo_21_0
13064                            <1>
13065                            <1> sysvideo_21_5:
13066                            <1>       ; write/set all DAC color registers
13067 00010154 89CE              <1>       mov    esi, ecx ; user's buffer address
13068 00010156 BF00600900        <1>       mov    edi, VBE3STACKADDR
13069 0001015B 89FB              <1>       mov    ebx, edi
13070 0001015D B900030000        <1>       mov    ecx, 768 ; 256*3
13071 00010162 E825140000        <1>       call   transfer_from_user_buffer
13072 00010167 7299              <1>       jc     short sysvideo_21_1
13073 00010169 890D[64030300]    <1>       mov    [u.r0], ecx ; actual transfer count
13074                            <1>
13075 0001016F 89DE              <1>       mov    esi, ebx ; VBE3STACKADDR
13076 00010171 66BAC803          <1>       mov    dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
13077 00010175 28C0              <1>       sub    al, al ; 0
13078 00010177 EE                <1>       out    dx, al
13079                            <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
13080 00010178 FEC2              <1>       inc    dl
13081                            <1> sysvideo_21_6:
13082 0001017A AC                <1>       lodsb
13083 0001017B EE                <1>       out    dx, al
13084 0001017C AC                <1>       lodsb
13085 0001017D EE                <1>       out    dx, al
13086 0001017E AC                <1>       lodsb
13087 0001017F EE                <1>       out    dx, al
13088 00010180 E2F8              <1>       loop   sysvideo_21_6
13089 00010182 EB30              <1>       jmp    short sysvideo_21_9
13090                            <1>
13091                            <1> sysvideo_21_7:
13092                            <1>       ; BMP file type palette handling
13093                            <1>
13094 00010184 F6C301            <1>       test   bl, 1
13095 00010187 756E              <1>       jnz    short sysvideo_21_12 ; set/write DAC colors
13096                            <1>
13097                            <1>       ; Read DAC color register or all DAC color registers
13098 00010189 F6C302            <1>       test   bl, 2  ; read single DAC color register
13099 0001018C 742B              <1>       jz     short sysvideo_21_10 ; read all DAC color regs
13100                            <1>
13101                            <1>       ; read single DAC color register
13102                            <1>       ; CL = DAC color register (index)
13103                            <1>
13104 0001018E 66BAC703          <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
13105 00010192 88C8              <1>       mov    al, cl ; DAC color register
13106 00010194 31C9              <1>       xor    ecx, ecx
13107 00010196 EE                <1>       out    dx, al
13108                            <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
13109 00010197 B2C9              <1>       mov    dl, 0C9h
13110 00010199 EC                <1>       in     al, dx
13111 0001019A C0E002            <1>       shl    al, 2
13112 0001019D 88C5              <1>       mov    ch, al ; red
13113 0001019F EC                <1>       in     al, dx
13114 000101A0 C0E002            <1>       shl    al, 2
13115 000101A3 88C1              <1>       mov    cl, al ; green
13116 000101A5 EC                <1>       in     al, dx
13117 000101A6 C0E002            <1>       shl    al, 2
13118                            <1>       ; 21/02/2021
13119 000101A9 C1E108            <1>       shl    ecx, 8
13120 000101AC 88C1              <1>       mov    cl, al ; blue
13121                            <1>       ; CL = Blue, CH = Green, byte 3 = Red, byte 4 = 0
13122                            <1> sysvideo_21_8:
13123 000101AE 890D[64030300]    <1>       mov    [u.r0], ecx
13124                            <1> sysvideo_21_9:
13125 000101B4 E917D2FFFF        <1>       jmp    sysret
13126                            <1> sysvideo_21_10:
13127                            <1>       ; read all DAC color registers
13128 000101B9 89CD              <1>       mov    ebp, ecx ; user's buffer address
```

```
13129 000101BB BF00600900          <1>       mov   edi, VBE3STACKADDR
13130 000101C0 89FE                <1>       mov   esi, edi
13131 000101C2 B900040000          <1>       mov   ecx, 1024 ; 256*4
13132 000101C7 51                  <1>       push  ecx
13133 000101C8 66BAC703            <1>       mov   dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
13134 000101CC 28C0                <1>       sub   al, al ; 0
13135 000101CE EE                  <1>       out   dx, al
13136                              <1>       ;mov  dx, 3C9h ; VGAREG_DAC_DATA
13137 000101CF B2C9                <1>       mov   dl, 0C9h
13138                              <1> sysvideo_21_11:
13139 000101D1 31DB                <1>       xor   ebx, ebx
13140 000101D3 EC                  <1>       in    al, dx ; Red
13141 000101D4 C0E002              <1>       shl   al, 2
13142 000101D7 88C7                <1>       mov   bh, al
13143 000101D9 EC                  <1>       in    al, dx ; Green
13144 000101DA C0E002              <1>       shl   al, 2
13145 000101DD 88C3                <1>       mov   bl, al
13146 000101DF EC                  <1>       in    al, dx ; Blue
13147 000101E0 C0E002              <1>       shl   al, 2
13148 000101E3 C1E308              <1>       shl   ebx, 8
13149 000101E6 89D8                <1>       mov   eax, ebx ; 00RRGGBBh
13150 000101E8 AB                  <1>       stosd
13151 000101E9 E2E6                <1>       loop  sysvideo_21_11
13152 000101EB 59                  <1>       pop   ecx
13153                              <1>
13154 000101EC 89EF                <1>       mov   edi, ebp ; user's buffer address
13155                              <1>       ;mov  esi, VBE3STACKADDR
13156                              <1>       ;mov  ecx, 1024 = 4*256
13157 000101EE E84F130000          <1>       call  transfer_to_user_buffer
13158 000101F3 72BF                <1>       jc    short sysvideo_21_9
13159                              <1>       ;mov  [u.r0], ecx ; actual transfer count
13160 000101F5 EBB7                <1>       jmp   short sysvideo_21_8
13161                              <1>
13162                              <1> sysvideo_21_12:
13163                              <1>       ; Set/Write DAC color register or all registers
13164 000101F7 F6C302              <1>       test  bl, 2 ; write/set single DAC color register
13165 000101FA 7427                <1>       jz    short sysvideo_21_13 ; set all DAC color regs
13166                              <1>
13167                              <1>       ; set single DAC color register
13168                              <1>       ; CL = DAC color register (index)
13169                              <1>       ; (byte 1 = Blue, byte 2 = Green, byte 3 = Red)
13170                              <1>
13171 000101FC 66BAC803           <1>       mov   dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
13172 00010200 88C8                <1>       mov   al, cl ; DAC color register (index)
13173 00010202 88EC                <1>       mov   ah, ch ; Blue
13174 00010204 C1E910              <1>       shr   ecx, 16
13175 00010207 EE                  <1>       out   dx, al
13176                              <1>       ;mov  dx, 3C9h ; VGAREG_DAC_DATA
13177 00010208 FEC2                <1>       inc   dl
13178 0001020A 88E8                <1>       mov   al, ch ; Red
13179 0001020C C0E802              <1>       shr   al, 2
13180 0001020F EE                  <1>       out   dx, al
13181 00010210 88C8                <1>       mov   al, cl ; Green
13182 00010212 C0E802              <1>       shr   al, 2
13183 00010215 EE                  <1>       out   dx, al
13184 00010216 88E0                <1>       mov   al, ah ; Blue
13185 00010218 C0E802              <1>       shr   al, 2
13186 0001021B EE                  <1>       out   dx, al
13187                              <1>       ;rol  ecx, 8
13188 0001021C C1E108              <1>       shl   ecx, 8 ; 21/02/2021
13189 0001021F 88E1                <1>       mov   cl, ah
13190 00010221 EB8B                <1>       jmp   short sysvideo_21_8
13191                              <1>
13192                              <1> sysvideo_21_13:
13193                              <1>       ; write/set all DAC color registers
13194 00010223 89CE                <1>       mov   esi, ecx ; user's buffer address
13195 00010225 BF00600900          <1>       mov   edi, VBE3STACKADDR
13196 0001022A 89FB                <1>       mov   ebx, edi
13197 0001022C B900040000          <1>       mov   ecx, 1024 ; 256*4
13198 00010231 E856130000          <1>       call  transfer_from_user_buffer
13199 00010236 722E                <1>       jc    short sysvideo_21_15
13200 00010238 890D[64030300]      <1>       mov   [u.r0], ecx ; actual transfer count
13201                              <1>
13202 0001023E 89DE                <1>       mov   esi, ebx ; VBE3STACKADDR
13203 00010240 66BAC803            <1>       mov   dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
13204 00010244 28C0                <1>       sub   al, al ; 0
13205 00010246 EE                  <1>       out   dx, al
13206                              <1>       ;mov  dx, 3C9h ; VGAREG_DAC_DATA
13207 00010247 FEC2                <1>       inc   dl
13208                              <1> sysvideo_21_14:
13209 00010249 AD                  <1>       lodsd
13210                              <1>       ; byte 0 = Blue, byte 1 = Green, byte 2 = Red
13211                              <1>       ; 21/02/2021
13212 0001024A 89C3                <1>       mov   ebx, eax ; BL = Blue, BH = Green
13213 0001024C C1CB08              <1>       ror   ebx, 8 ; BL = Green, BH = Red
13214 0001024F 88F8                <1>       mov   al, bh
13215 00010251 C0E802              <1>       shr   al, 2
13216 00010254 EE                  <1>       out   dx, al ; Red
13217 00010255 88D8                <1>       mov   al, bl
13218 00010257 C0E802              <1>       shr   al, 2
13219 0001025A EE                  <1>       out   dx, al ; Green
13220 0001025B C1C308              <1>       rol   ebx, 8 ; BL = Blue
13221 0001025E 88D8                <1>       mov   al, bl
13222 00010260 C0E802              <1>       shr   al, 2
13223 00010263 EE                  <1>       out   dx, al ; Blue
13224 00010264 E2E3                <1>       loop  sysvideo_21_14
13225                              <1> sysvideo_21_15:
13226 00010266 E965D1FFFF          <1>       jmp   sysret
13227                              <1>
13228                              <1> sysvideo_22:
13229                              <1>       ; 28/02/2021
13230                              <1>       ; 22/01/2021
13231                              <1>       ; 17/01/2021
13232                              <1>       ; 04/12/2020
13233                              <1>       ; 03/12/2020
```

```
13234                                  <1>       ; BH = 9
13235                                  <1>       ; Set/Get VESA VBE3 protected mode interface params
13236                                  <1>
13237                                  <1>       ; 22/01/2021
13238                                  <1>       ;cmp   byte [vbe3], 3
13239                                  <1>       ;jne   short sysvideo_25 ; not applicable if
13240                                  <1>                                ; vbe3 compatible video bios
13241                                  <1>                                ; is not detected by kernel
13242 0001026B 80FB02                 <1>       cmp    bl, 2
13243                                  <1>       ;ja    short sysvideo_25 ; bl > 2 not implemented
13244                                  <1>       ; 17/01/2021
13245 0001026E 7716                   <1>       ja     short sysvideo_22_0 ; srvs flag sub function
13246                                  <1>       ;jb    short sysvideo_23
13247                                  <1>
13248                                  <1>       ; 21/01/2021
13249 00010270 803D[4E090000]03       <1>       cmp    byte [vbe3], 3
13250                                  <1>       ;jne   short sysvideo_25 ; not applicable if
13251                                  <1>                                ; vbe3 compatible video bios
13252                                  <1>                                ; is not detected by kernel
13253 00010277 75ED                   <1>       jne    short sysvideo_21_15 ; 28/02/2021
13254                                  <1>
13255 00010279 80FB01                 <1>       cmp    bl, 1
13256 0001027C 7673                   <1>       jna    short sysvideo_23
13257                                  <1>
13258 0001027E 8A1D[04120300]         <1>       mov    bl, [pmi32] ; Video bios 32 bit PMI functions
13259 00010284 EB78                   <1>       jmp    short sysvideo_24
13260                                  <1>
13261                                  <1> sysvideo_22_0:
13262                                  <1>       ; 17/01/2021
13263                                  <1>       ; save/restore video state user permission
13264 00010286 80FB05                 <1>       cmp    bl, 5
13265 00010289 771E                   <1>       ja     short sysvideo_22_2
13266 0001028B 7208                   <1>       jb     short sysvideo_22_1
13267                                  <1>       ; get srvs flag value/status
13268 0001028D 8A1D[68120300]         <1>       mov    bl, [srvsf] ; 0 = disabled, 1 = enabled
13269 00010293 EB2C                   <1>       jmp    short sysvideo_22_3
13270                                  <1>
13271                                  <1> sysvideo_22_1:
13272                                  <1>       ; permission (root and multi tasking) check
13273 00010295 E836000000             <1>       call   sysvideo_22_4
13274 0001029A 736A                   <1>       jnc    short sysvideo_25 ; not permitted !
13275                                  <1>       ; cf = 1
13276 0001029C 80EB03                 <1>       sub    bl, 3 ; disable = 0, enable = 1
13277                                  <1>       ; 22/01/2021
13278 0001029F 881D[68120300]         <1>       mov    [srvsf], bl
13279 000102A5 FEC3                   <1>       inc    bl ; 1 = disabled, 2 = enabled
13280 000102A7 EB18                   <1>       jmp    short sysvideo_22_3
13281                                  <1>
13282                                  <1> sysvideo_22_2:
13283 000102A9 80FB06                 <1>       cmp    bl, 6
13284                                  <1>       ;ja    short sysvideo_25 ; invalid/unimplemented
13285                                  <1>       ; 28/02/2021
13286 000102AC 7733                   <1>       ja     short sysvideo_22_6
13287                                  <1>       ; get VESA VBE number/status
13288 000102AE 8A25[4E090000]         <1>       mov    ah, [vbe3] ; vbe3 = 3, vbe2 = 2, others = 0
13289 000102B4 A0[4F090000]           <1>       mov    al, [vbe2bios] ; bochs/qemu/vbox emulator status
13290 000102B9 66A3[64030300]         <1>       mov    [u.r0], ax
13291 000102BF EB45                   <1>       jmp    short sysvideo_25
13292                                  <1>
13293                                  <1> sysvideo_22_3:
13294                                  <1>       ; 22/01/2021
13295 000102C1 8A3D[69120300]         <1>       mov    bh, [srvso] ; state options (> 80h -> svga)
13296 000102C7 66891D[64030300]       <1>       mov    [u.r0], bx ; function result is return value
13297 000102CE EB36                   <1>       jmp    short sysvideo_25
13298                                  <1>
13299                                  <1> sysvideo_22_4:
13300                                  <1>       ; 17/01/2021 - permission will be given by root only
13301 000102D0 803D[3A8D0100]00       <1>       cmp    byte [multi_tasking], 0 ; in single user mode
13302 000102D7 7707                   <1>       ja     short sysvideo_22_5
13303                                  <1>       ; 19/01/2021
13304 000102D9 803D[B0030300]01       <1>       cmp    byte [u.uid], 1 ; ([u.uid] = 0 -> root)
13305                                  <1> sysvideo_22_5:
13306                                  <1>       ; [multi_tasking] = 0 & [u.uid] = 0 -> CF = 1
13307                                  <1>       ; otherwise -> CF = 0
13308 000102E0 C3                     <1>       retn
13309                                  <1>
13310                                  <1> sysvideo_22_6:
13311                                  <1>       ; 28/02/2021
13312 000102E1 80FB09                 <1>       cmp    bl, 9
13313 000102E4 7720                   <1>       ja     short sysvideo_25 ; invalid/unimplemented
13314 000102E6 7436                   <1>       je     short sysvideo_22_9
13315 000102E8 80FB08                 <1>       cmp    bl, 8
13316 000102EB 721E                   <1>       jb     short sysvideo_22_7
13317                                  <1>
13318                                  <1>       ; BL = 8
13319                                  <1>       ; Set default true color bpp to 24
13320                                  <1>
13321 000102ED B318                   <1>       mov    bl, 24
13322                                  <1>       ;mov   [truecolor], al     ; 24bpp (RRGGBBh)
13323                                  <1>       ;mov   [u.r0], al
13324                                  <1>       ;jmp   short sysvideo_25
13325 000102EF EB25                   <1>       jmp    short sysvideo_22_8
13326                                  <1>
13327                                  <1> sysvideo_23:
13328                                  <1>       ; 17/01/2021
13329                                  <1>       ; permission (root and multi tasking) check
13330 000102F1 E8DAFFFFFF             <1>       call   sysvideo_22_4
13331 000102F6 730E                   <1>       jnc    short sysvideo_25 ; not permitted !
13332                                  <1>
13333 000102F8 881D[04120300]         <1>       mov    [pmi32], bl ; 1 = enabled, 0 = disabled
13334                                  <1> sysvideo_24:
13335 000102FE FEC3                   <1>       inc    bl
13336                                  <1> sysvideo_22_10:    ; 28/02/2021
13337 00010300 881D[64030300]         <1>       mov    [u.r0], bl ; function result is return value
13338                                  <1> sysvideo_25:
```

```
13339 00010306 E9C5D0FFFF          <1>       jmp    sysret
13340                              <1>
13341                              <1> sysvideo_22_7:
13342                              <1>       ; BL = 7
13343                              <1>       ; Set default true color bpp to 32
13344                              <1>       ; (it will set if [VBE3]=3)
13345                              <1>
13346                              <1>       ; Note: This sub function is used to set 24bpp
13347                              <1>       ; VESA VBE video modes to 32bpp.. because,
13348                              <1>       ; old hardware uses 24 bpp but new video hardware
13349                              <1>       ; uses 32bpp for same VESA VBE truecolor modes.
13350                              <1>       ; (For example: VBE mode 112h is 640*480, 24bpp but
13351                              <1>       ; new hardware uses/apply it as 640*480, 32bpp.)
13352                              <1>       ; So, TRDOS 386 v2.0.3 kernel will check [truecolor]
13353                              <1>       ; status is 32 bpp or not and it will change 24bpp
13354                              <1>       ; to 32bpp if default [truecolor] value is 32, for
13355                              <1>       ; same video mode number.
13356                              <1>
13357 0001030B 803D[4E090000]03   <1>       cmp    byte [vbe3], 3
13358 00010312 75F2               <1>       jne    short sysvideo_25 ; Only applicable
13359                              <1>                                ; for VBE3 video hardware!
13360 00010314 B320               <1>       mov    bl, 32
13361                              <1> sysvideo_22_8:
13362 00010316 881D[6F7D0100]     <1>       mov    [truecolor], bl     ; 32bpp (00RRGGBBh)
13363                              <1>       ;mov   [u.r0], bl
13364                              <1>       ;jmp   short sysvideo_25
13365 0001031C EBE2               <1>       jmp    short sysvideo_22_10
13366                              <1>
13367                              <1> sysvideo_22_9:
13368                              <1>       ; BL = 9
13369                              <1>       ; Return default true color bpp
13370 0001031E 8A1D[6F7D0100]     <1>       mov    bl, [truecolor]
13371 00010324 EBDA               <1>       jmp    short sysvideo_22_10
13372                              <1> ;sysvideo_22_10:
13373                              <1>       ;mov   [u.r0], bl
13374                              <1>       ;jmp   sysret
13375                              <1>
13376                              <1> sysvideo_26:
13377                              <1>       ; 23/12/2020
13378                              <1>       ; BH = 10
13379                              <1>       ; Map video memory to user's buffer
13380                              <1>       ; (multiuser/owner r/w permisions are ignored
13381                              <1>       ;  for current TRDOS 386 version !)
13382                              <1>
13383 00010326 6681E100F0         <1>       and    cx, ~4095  ; clear low 12 bits
13384 0001032B 09C9               <1>       or     ecx, ecx ; start address of user's buffer
13385 0001032D 74D7               <1>       jz     short sysvideo_25 ; error !
13386                              <1>
13387 0001032F 80FB01             <1>       cmp    bl, 1  ; VGA memory mapping ?
13388 00010332 740E               <1>       je     short sysvideo_26_1
13389 00010334 7718               <1>       ja     short sysvideo_26_2
13390                              <1> sysvideo_26_0:
13391                              <1>       ; BL = 0 : CGA memory (0B8000h) map (32K)
13392 00010336 B800800B00         <1>       mov    eax, 0B8000h
13393 0001033B BB00800000         <1>       mov    ebx, 32768
13394 00010340 EB37               <1>       jmp    short sysvideo_26_3
13395                              <1> sysvideo_26_1:
13396                              <1>       ; BL = 1 : VGA memory (0A0000h) map (64K)
13397 00010342 B800000A00         <1>       mov    eax, 0A0000h
13398 00010347 BB00000100         <1>       mov    ebx, 65536
13399 0001034C EB2B               <1>       jmp    short sysvideo_26_3
13400                              <1> sysvideo_26_2:
13401                              <1>       ; BL = 2 : SVGA memory (LFB) map to user's buffer
13402 0001034E 803D[4E090000]02   <1>       cmp    byte [vbe3], 2  ; VESA VBE 2/3 vbios ready ?
13403 00010355 72AF               <1>       jb     short sysvideo_25  ; no, error !
13404 00010357 6681E200F0         <1>       and    dx, ~4095 ; clear low 12 bits
13405 0001035C 09D2               <1>       or     edx, edx  ; buffer size in bytes
13406 0001035E 74A6               <1>       jz     short sysvideo_25  ; error
13407 00010360 89D3               <1>       mov    ebx, edx
13408 00010362 A1[14120300]       <1>       mov    eax, [LFB_ADDR] ; [LFB_Info+LFBINFO.LFB_addr]
13409 00010367 21C0               <1>       and    eax, eax
13410 00010369 7425               <1>       jz     short sysvideo_26_5
13411                              <1>                              ; (LFB parms are not set yet)
13412 0001036B 3B1D[18120300]     <1>       cmp    ebx, [LFB_SIZE] ; [LFB_Info+LFBINFO.LFB_size]
13413 00010371 7606               <1>       jna    short sysvideo_26_3
13414 00010373 8B1D[18120300]     <1>       mov    ebx, [LFB_SIZE]
13415                              <1> sysvideo_26_3:
13416 00010379 52                 <1>       push   edx
13417 0001037A 53                 <1>       push   ebx    ; buffer size in bytes
13418 0001037B 51                 <1>       push   ecx    ; user's buffer address
13419 0001037C 87D9               <1>       xchg   ebx, ecx
13420 0001037E C1E90C             <1>       shr    ecx, 12 ; convert buffer size to page count
13421 00010381 E88E5EFFFF         <1>       call   direct_memory_access
13422 00010386 59                 <1>       pop    ecx    ; user's buffer address
13423 00010387 5B                 <1>       pop    ebx    ; buffer size
13424 00010388 5A                 <1>       pop    edx
13425                              <1>       ;jc    short sysvideo_25 ; error !
13426                              <1>                              ; [u.r0] = 0
13427                              <1>       ; 28/02/2021
13428 00010389 7234               <1>       jc     short sysvideo_27_0 ; error !
13429                              <1>
13430                              <1> ;sysvideo_26_4:
13431                              <1>       ;mov   ebp, [u.usp] ; ebp points to user's registers
13432                              <1>       ;mov   [ebp+20], edx ; return to user with EDX value
13433                              <1>       ;mov   [ebp+16], ebx ; EBX
13434                              <1>       ;mov   [ebp+24], ecx ; ECX
13435                              <1>       ; eax = physical address of video memory (LFB)
13436                              <1>       ;mov   [u.r0], eax
13437                              <1>       ;jmp   sysret
13438 0001038B E919FCFFFF         <1>       jmp    sysvideo_26_4
13439                              <1>
13440                              <1> sysvideo_26_5:
13441 00010390 66A1[D90E0000]     <1>       mov    ax, [def_LFB_addr] ; default LFB for mode 118h
13442                              <1>       ; ah must be 0C0h or 0D0h or E0h
13443                              <1>       ; others are nonsence !?
```

```
13444 00010396 08E4               <1>       or    ah, ah
13445                             <1>       ;jz   short sysvideo_25 ; invalid lfb addr or
13446                             <1>                               ; it is not a vbe2 -bochs emu-
13447                             <1>                               ; or vbe3 -real- video bios
13448                             <1>       ; 28/02/2021
13449 00010398 7425               <1>       jz    short sysvideo_27_0 ; invalid LFB address
13450                             <1>
13451 0001039A 80FCF0             <1>       cmp   ah, 0F0h
13452                             <1>       ;jnb  short sysvideo_25 ; nonsence !?
13453                             <1>       ; 28/02/2021
13454 0001039D 7320               <1>       jnb   short sysvideo_27_0 ; nonsence !?
13455                             <1>
13456 0001039F C1E010             <1>       shl   eax, 16
13457                             <1>       ;jz   short sysvideo_25 ; eax = 0
13458                             <1>
13459 000103A2 81FB00907E00       <1>       cmp   ebx, 1920*1080*4 ; maximum value of possible
13460                             <1>                              ; buffer sizes
13461 000103A8 76CF               <1>       jna   short sysvideo_26_3  ; buffer size is proper
13462                             <1>       ; resize buffer to fit 4GB limit
13463 000103AA BB00907E00         <1>       mov   ebx, 1920*1080*4
13464 000103AF EBC8               <1>       jmp   short sysvideo_26_3
13465                             <1>
13466                             <1> sysvideo_27:
13467                             <1>       ; 16/02/2021
13468                             <1>       ; 18/01/2021
13469 000103B1 80FF0C             <1>       cmp   bh, 12
13470 000103B4 0F8778010000       <1>       ja    sysvideo_28 ; 19/01/2021
13471                             <1>
13472                             <1>       ; BH = 12
13473                             <1>       ; Font sub functions.
13474                             <1>       ; 12/02/2021
13475                             <1>       ; 11/01/2021
13476                             <1>       ; 10/01/2021
13477                             <1>       ;   BL = 0 : Disable system font overwrite
13478                             <1>       ;   BL = 1 : Enable system font overwrite
13479                             <1>       ;   BL = 2 : Read system font 8x8
13480                             <1>       ;   BL = 3 : Read system font 8x14
13481                             <1>       ;   BL = 4 : Read system font 8x16
13482                             <1>       ;   BL = 5 : Read user defined font 8x8
13483                             <1>       ;   BL = 6 : Read user defined font 8x16
13484                             <1>       ;   BL = 7 : Write system font 8x8
13485                             <1>       ;   BL = 8 : Write system font 8x14
13486                             <1>       ;   BL = 9 : Write system font 8x16
13487                             <1>       ;   BL = 10 : Write user defined font 8x8
13488                             <1>       ;   BL = 11 : Write user defined font 8x16
13489                             <1>       ;
13490                             <1>       ;  BL > 11 : invalid (not implemented)
13491                             <1>       ;
13492                             <1>       ; For BL = 1 to 11
13493                             <1>       ;   ECX = number of characters (<= 256)
13494                             <1>       ;   EDX = first character (ascii code in DL)
13495                             <1>       ;   ESI = user's buffer address
13496                             <1>       ;
13497                             <1>       ; Return: EAX = character count
13498                             <1>
13499 000103BA 80FB0B             <1>       cmp   bl, 11
13500 000103BD 7605               <1>       jna   short sysvideo_27_1
13501                             <1> sysvideo_27_0:
13502 000103BF E90CD0FFFF         <1>       jmp   sysret ; not implemented yet !
13503                             <1> sysvideo_27_1:
13504 000103C4 66B80001           <1>       mov   ax, 256
13505 000103C8 08DB               <1>       or    bl, bl
13506 000103CA 750E               <1>       jnz   short sysvideo_27_3
13507                             <1>
13508                             <1>       ; bl = 0
13509                             <1>       ; disable system font overwrite
13510                             <1>
13511 000103CC 8025[66120300]7F   <1>       and   byte [ufont], 7Fh ; clear bit 7
13512                             <1> sysvideo_27_2:
13513                             <1>       ;mov  word [u.r0], 256  ; > 0 -> successful
13514 000103D3 A3[64030300]       <1>       mov   [u.r0], eax ; 256
13515 000103D8 EBE5               <1>       jmp   short sysvideo_27_0
13516                             <1> sysvideo_27_3:
13517 000103DA 80FB01             <1>       cmp   bl, 1
13518 000103DD 7710               <1>       ja    short sysvideo_27_4
13519                             <1>
13520                             <1>       ; bl = 1
13521                             <1>       ; enable system font overwrite
13522                             <1>       ;     if [multi_tasking]= 0 and [u.uid] = 0
13523                             <1>
13524                             <1>       ;cmp  byte [multi_tasking], 0
13525                             <1>       ;                 ; multi tasking enabled ?
13526                             <1>       ;ja   short sysvideo_27_0 ; yes
13527                             <1>       ;; 19/01/2021
13528                             <1>       ;; system maintenance or single user mode
13529                             <1>       ;cmp  byte [u.uid], 0  ; root ?
13530                             <1>       ;ja   short sysvideo_27_0 ; no
13531                             <1>
13532                             <1>       ; 19/01/2021
13533                             <1>       ; multi tasking & root check
13534 000103DF E8ECFEFFFF         <1>       call  sysvideo_22_4
13535 000103E4 73D9               <1>       jnc   short sysvideo_27_0 ; not permitted
13536                             <1>
13537                             <1>       ; [multi_tasking]= 0 and [u.uid] = 0
13538                             <1>
13539 000103E6 800D[66120300]80   <1>       or    byte [ufont], 80h ; set bit 7
13540                             <1>
13541 000103ED EBE4               <1>       jmp   short sysvideo_27_2
13542                             <1>
13543                             <1> sysvideo_27_4:
13544 000103EF 09C9               <1>       or    ecx, ecx
13545 000103F1 74CC               <1>       jz    short sysvideo_27_0
13546 000103F3 21D2               <1>       and   edx, edx
13547 000103F5 7410               <1>       jz    short sysvideo_27_4_0
13548                             <1>       ;mov  ax, 256
```

```
13549 000103F7 39C1                <1>       cmp    ecx, eax ; 256
13550 000103F9 77C4                <1>       ja     short sysvideo_27_0
13551 000103FB 48                  <1>       dec    eax
13552 000103FC 39C2                <1>       cmp    edx, eax ; 255
13553 000103FE 77BF                <1>       ja     short sysvideo_27_0
13554 00010400 40                  <1>       inc    eax
13555 00010401 29D0                <1>       sub    eax, edx ; 256 - DX
13556 00010403 39C8                <1>       cmp    eax, ecx
13557 00010405 72B8                <1>       jb     short sysvideo_27_0
13558                              <1>
13559                              <1> sysvideo_27_4_0:
13560 00010407 89F5                <1>       mov    ebp, esi
13561                              <1>
13562 00010409 80FB06              <1>       cmp    bl, 6
13563 0001040C 776C                <1>       ja     short sysvideo_27_13
13564 0001040E 7210                <1>       jb     short sysvideo_27_5
13565                              <1>       ; bl = 6
13566 00010410 F605[66120300]10    <1>       test   byte [ufont], 16 ; 8x16 user font loaded ?
13567 00010417 74A6                <1>       jz     short sysvideo_27_0
13568                              <1>       ; read 8x16 user defined font
13569 00010419 BE00400900          <1>       mov    esi, VGAFONT16USER
13570 0001041E EB0C                <1>       jmp    short sysvideo_27_6
13571                              <1> sysvideo_27_5:
13572 00010420 80FB04              <1>       cmp    bl, 4
13573 00010423 723D                <1>       jb     short sysvideo_27_11
13574 00010425 7723                <1>       ja     short sysvideo_27_9
13575                              <1>       ; bl = 4
13576                              <1>       ; read 8x16 system font
13577 00010427 BE[3C6D0100]        <1>       mov    esi, vgafont16
13578                              <1> sysvideo_27_6:
13579                              <1>       ; read 8x16 font
13580 0001042C 66C1E204            <1>       shl    dx, 4 ; * 16
13581 00010430 66C1E104            <1>       shl    cx, 4 ; * 16 ; 16 bytes per char
13582                              <1> sysvideo_27_7:
13583 00010434 89EF                <1>       mov    edi, ebp
13584                              <1>       ;add   edi, edx ; 16/02/2021
13585 00010436 01D6                <1>       add    esi, edx
13586                              <1>       ; ecx = byte count
13587                              <1>       ; esi = source (in system memory)
13588                              <1>       ; edi = destination (in user memory)
13589 00010438 E805110000          <1>       call   transfer_to_user_buffer
13590 0001043D 7206                <1>       jc     short sysvideo_27_8
13591 0001043F 890D[64030300]      <1>       mov    [u.r0], ecx
13592                              <1> sysvideo_27_8:
13593 00010445 E986CFFFFF          <1>       jmp    sysret
13594                              <1> sysvideo_27_9:
13595                              <1>       ; bl = 5
13596 0001044A F605[66120300]08    <1>       test   byte [ufont], 8 ; 8x8 user font loaded ?
13597 00010451 74F2                <1>       jz     short sysvideo_27_8
13598                              <1>       ; read 8x8 user defined font
13599 00010453 BE00500900          <1>       mov    esi, VGAFONT8USER
13600                              <1> sysvideo_27_10:
13601                              <1>       ; read 8x8 font
13602 00010458 66C1E203            <1>       shl    dx, 3 ; * 8
13603 0001045C 66C1E103            <1>       shl    cx, 3 ; * 8 ; 8 bytes per char
13604 00010460 EBD2                <1>       jmp    short sysvideo_27_7
13605                              <1>
13606                              <1> sysvideo_27_11:
13607 00010462 80FB03              <1>       cmp    bl, 3  ; 8x14 system font
13608 00010465 720C                <1>       jb     short sysvideo_27_12 ; 8x8 system font
13609                              <1>       ; bl = 3
13610                              <1>       ; read 8x14 system font
13611                              <1>       ;mov   al, 14
13612                              <1>       ;mul   dl
13613                              <1>       ;mov   dx, ax
13614                              <1>       ;push  edx
13615                              <1>       ;mov   ax, 14
13616                              <1>       ;mul   cx
13617                              <1>       ;mov   cx, ax
13618                              <1>       ;pop   edx
13619 00010467 E8A9000000          <1>       call   sysvideo_27_14
13620 0001046C BE[3C5F0100]        <1>       mov    esi, vgafont14
13621 00010471 EBC1                <1>       jmp    short sysvideo_27_7
13622                              <1>
13623                              <1> sysvideo_27_12:
13624                              <1>       ; bl = 2
13625                              <1>       ; read 8x8 system font
13626 00010473 BE[3C570100]        <1>       mov    esi, vgafont8
13627 00010478 EBDE                <1>       jmp    short sysvideo_27_10
13628                              <1>
13629                              <1> sysvideo_27_13:
13630                              <1>       ; overwrite font
13631 0001047A 80FB0A              <1>       cmp    bl, 10
13632 0001047D 7772                <1>       ja     short sysvideo_27_22 ; 8x16 user font
13633 0001047F 7224                <1>       jb     short sysvideo_27_15
13634                              <1>       ; bl = 10
13635 00010481 BF00500900          <1>       mov    edi, VGAFONT8USER
13636 00010486 F605[66120300]08    <1>       test   byte [ufont], 8 ; 8x8 user font loaded ?
13637 0001048D 7558                <1>       jnz    short sysvideo_27_21 ; yes
13638 0001048F 08ED                <1>       or     ch, ch ; cx = 256
13639                              <1>       ;jnz   short sysvideo_27_21 ; 256 chars
13640 00010491 7406                <1>       jz     short sysvideo_27_13_0
13641 00010493 66B90008            <1>       mov    cx, 8*256
13642 00010497 EB37                <1>       jmp    short sysvideo_27_18_0
13643                              <1> sysvideo_27_13_0:
13644                              <1>       ; copy system font to user font before overwrite
13645 00010499 BE[3C570100]        <1>       mov    esi, vgafont8
13646                              <1>       ;push  edi
13647                              <1>       ;push  ecx
13648                              <1>       ;mov   cl, 64
13649                              <1>       ;rep   movsd
13650                              <1>       ;pop   ecx
13651                              <1>       ;pop   edi
13652                              <1>       ;mov   esi, ebp ; user's font buffer
13653 0001049E E884000000          <1>       call   sysvideo_27_23
```

```
13654 000104A3 EB42                <1>       jmp    short sysvideo_27_21
13655                              <1>
13656                              <1> sysvideo_27_15:
13657                              <1>       ; check system font overwrite permission
13658 000104A5 F605[66120300]80    <1>       test   byte [ufont], 80h
13659 000104AC 7497                <1>       jz     short sysvideo_27_8
13660                              <1>
13661 000104AE 80FB08              <1>       cmp    bl, 8
13662 000104B1 773E                <1>       ja     short sysvideo_27_22 ; 8x16 system font
13663 000104B3 722D                <1>       jb     short sysvideo_27_20 ; 8x8 system font
13664                              <1>       ; bl = 8
13665                              <1>       ; overwrite 8x14 system font
13666                              <1>       ;mov   al, 14
13667                              <1>       ;mul   dl
13668                              <1>       ;mov   dx, ax
13669                              <1>       ;push  edx
13670                              <1>       ;mov   ax, 14
13671                              <1>       ;mul   cx
13672                              <1>       ;mov   cx, ax
13673                              <1>       ;pop   edx
13674 000104B5 E85B000000         <1>       call   sysvideo_27_14
13675 000104BA BF[3C5F0100]        <1>       mov    edi, vgafont14
13676 000104BF EB0D                <1>       jmp    short sysvideo_27_18
13677                              <1> sysvideo_27_16:
13678                              <1>       ; bl = 9
13679                              <1>       ; overwrite 8x16 system font
13680 000104C1 BF[3C6D0100]        <1>       mov    edi, vgafont16
13681                              <1> sysvideo_27_17:
13682                              <1>       ; overwrite 8x16 font
13683 000104C6 66C1E204            <1>       shl    dx, 4 ; * 16
13684 000104CA 66C1E104            <1>       shl    cx, 4 ; * 16 ; 16 bytes per char
13685                              <1> sysvideo_27_18:
13686 000104CE 01D7                <1>       add    edi, edx
13687                              <1>       ;add   esi, edx ; 16/02/2021
13688                              <1> sysvideo_27_18_0:
13689                              <1>       ; ecx = byte count
13690                              <1>       ; esi = source (in user memory)
13691                              <1>       ; edi = destination (in system memory)
13692 000104D0 E8B7100000         <1>       call   transfer_from_user_buffer
13693 000104D5 7206                <1>       jc     short sysvideo_27_19
13694 000104D7 890D[64030300]      <1>       mov    [u.r0], ecx
13695                              <1> sysvideo_27_19:
13696 000104DD E9EECEFFFF          <1>       jmp    sysret
13697                              <1> sysvideo_27_20:
13698                              <1>       ; bl = 7
13699                              <1>       ; overwrite 8x8 system font
13700 000104E2 BF[3C570100]        <1>       mov    edi, vgafont8
13701                              <1> sysvideo_27_21:
13702                              <1>       ; write 8x8 font
13703 000104E7 66C1E203            <1>       shl    dx, 3 ; * 8
13704 000104EB 66C1E103            <1>       shl    cx, 3 ; * 8 ; 8 bytes per char
13705 000104EF EBDD                <1>       jmp    short sysvideo_27_18
13706                              <1> sysvideo_27_22:
13707                              <1>       ; bl = 11
13708                              <1>       ; overwrite 8x16 user defined font
13709 000104F1 BF00400900          <1>       mov    edi, VGAFONT16USER
13710 000104F6 F605[66120300]10    <1>       test   byte [ufont], 16 ; 8x16 user font loaded ?
13711 000104FD 75C7                <1>       jnz    short sysvideo_27_17 ; yes
13712 000104FF 08ED                <1>       or     ch, ch ; cx = 256
13713                              <1>       ;jnz   short sysvideo_27_17 ; 256 chars
13714 00010501 7406                <1>       jz     short sysvideo_27_22_0
13715 00010503 66B90010            <1>       mov    cx, 16*256
13716 00010507 EBC7                <1>       jmp    short sysvideo_27_18_0
13717                              <1> sysvideo_27_22_0:
13718                              <1>       ; copy system font to user font before overwrite
13719 00010509 BE[3C6D0100]        <1>       mov    esi, vgafont16
13720                              <1>       ;push  edi
13721                              <1>       ;push  ecx
13722                              <1>       ;mov   cl, 64
13723                              <1>       ;rep   movsd
13724                              <1>       ;pop   ecx
13725                              <1>       ;pop   edi
13726                              <1>       ;mov   esi, ebp ; user's font buffer
13727 0001050E E814000000          <1>       call   sysvideo_27_23
13728 00010513 EBB1                <1>       jmp    short sysvideo_27_17
13729                              <1>
13730                              <1> sysvideo_27_14:
13731                              <1>       ; 16/02/2021
13732 00010515 52                  <1>       push   edx
13733 00010516 66B80E00            <1>       mov    ax, 14
13734 0001051A 66F7E1              <1>       mul    cx
13735 0001051D 89C1                <1>       mov    ecx, eax
13736 0001051F 5A                  <1>       pop    edx
13737 00010520 B00E                <1>       mov    al, 14
13738 00010522 F6E2                <1>       mul    dl
13739 00010524 89C2                <1>       mov    edx, eax
13740 00010526 C3                  <1>       retn
13741                              <1>
13742                              <1>       ;mov   al, 14
13743                              <1>       ;mul   dl
13744                              <1>       ;mov   dx, ax
13745                              <1>       ;push  edx
13746                              <1>       ;; 12/02/2021
13747                              <1>       ;mov   ax, 14
13748                              <1>       ;;mov  eax, 14
13749                              <1>       ;;mul  cx
13750                              <1>       ;mul   ecx
13751                              <1>       ;;mov  cx, ax
13752                              <1>       ;mov   ecx, eax
13753                              <1>       ;pop   edx
13754                              <1>       ;retn
13755                              <1>
13756                              <1> sysvideo_27_23:
13757 00010527 57                  <1>       push   edi
13758 00010528 51                  <1>       push   ecx
```

```
13759 00010529 B140                <1>       mov    cl, 64
13760 0001052B F3A5                <1>       rep    movsd
13761 0001052D 59                  <1>       pop    ecx
13762 0001052E 5F                  <1>       pop    edi
13763 0001052F 89EE                <1>       mov    esi, ebp ; user's font buffer
13764 00010531 C3                  <1>       retn
13765                              <1>
13766                              <1> sysvideo_28:
13767                              <1>       ; 24/01/2021
13768                              <1>       ; 23/01/2021
13769                              <1>       ; 18/01/2021
13770 00010532 80FF0E              <1>       cmp    bh, 14
13771 00010535 0F8275010000        <1>       jb     sysvideo_29
13772 0001053B 0F8754020000        <1>       ja     sysvideo_30
13773                              <1>
13774                              <1>       ; BH = 14
13775                              <1>       ; Save/Restore Super VGA video state
13776                              <1>
13777                              <1>       ; BL = options
13778                              <1>       ;      bit 0 - Save (0) or Restore (1)
13779                              <1>       ;      bit 1 - controller hardware state
13780                              <1>       ;      bit 2 - BIOS data state
13781                              <1>       ;      bit 3 - DAC state
13782                              <1>       ;      bit 4 - (extended) Register state
13783                              <1>       ;      bit 5 - system (0) or user (1) memory
13784                              <1>       ;      bit 6 - verify without transfer
13785                              <1>       ;      bit 7 - not used (must be 0)
13786                              <1>
13787                              <1>       ; ECX = Buffer address or VideoStateID
13788                              <1>
13789 00010541 803D[4E090000]02    <1>       cmp    byte [vbe3], 2 ;  VESA VBE2 or VBE3 ?
13790 00010548 7717                <1>       ja     short sysvideo_28_0 ; yes
13791 0001054A 7210                <1>       jb     short sysvideo_28_16 ; not a SVGA sys !
13792                              <1>
13793                              <1>       ; == VBE2 ==
13794                              <1>       ; Check Bochs/Qemu/VirtualBox PC emulator
13795                              <1>       ; (vbe2 is usable only for emulator's vbios)
13796 0001054C 8A25[4F090000]      <1>       mov    ah, [vbe2bios]
13797 00010552 80FCC0              <1>       cmp    ah, 0C0h
13798 00010555 7205                <1>       jb     short sysvideo_28_16 ; unknown vbios !
13799 00010557 80FCC5              <1>       cmp    ah, 0C5h
13800 0001055A 7605                <1>       jna    short sysvideo_28_0
13801                              <1>              ; Use kernel's vbios functions (video.s)
13802                              <1> sysvideo_28_16:
13803                              <1>       ; unknown vbios !
13804 0001055C E96FCEFFFF          <1>       jmp    sysret
13805                              <1>
13806                              <1> sysvideo_28_0:
13807 00010561 80FB7F              <1>       cmp    bl, 7Fh
13808 00010564 77F6                <1>       ja     short sysvideo_28_16 ; unknown options
13809                              <1>
13810 00010566 88DA                <1>       mov    dl, bl
13811 00010568 80E21F              <1>       and    dl, 1Fh
13812 0001056B D0EA                <1>       shr    dl, 1
13813 0001056D 74ED                <1>       jz     short sysvideo_28_16 ; invalid !
13814                              <1>       ; DL = VBE Function 4F04h Save/Restore options
13815                              <1>       ;  bit 0 : controller hardware state
13816                              <1>       ;  bit 1 : BIOS data state
13817                              <1>       ;  bit 2 : DAC state
13818                              <1>       ;  bit 3 : (extended) Register state
13819                              <1>
13820 0001056F F6C320              <1>       test   bl, 32 ; bit 5
13821 00010572 0F85B1000000        <1>       jnz    sysvideo_28_7 ; user buffer
13822                              <1>
13823                              <1>       ; source or destination is kernel/system buffer
13824                              <1>
13825 00010578 803D[68120300]00    <1>       cmp    byte [srvsf], 0 ; srs permission flag
13826 0001057F 76DB                <1>       jna    short sysvideo_28_16 ; not permitted
13827                              <1>
13828 00010581 F6C301              <1>       test   bl, 1
13829 00010584 743A                <1>       jz     short sysvideo_28_4 ; Save
13830                              <1>
13831                              <1>       ; Restore
13832 00010586 3B0D[6A120300]      <1>       cmp    ecx, [VideoStateID]
13833 0001058C 75CE                <1>       jne    short sysvideo_28_16 ; not correct ID !
13834                              <1>
13835 0001058E 0FB6CA              <1>       movzx  ecx, dl
13836 00010591 80CA80              <1>       or     dl, 80h
13837 00010594 3A15[69120300]      <1>       cmp    dl, [srvso]
13838 0001059A 75C0                <1>       jne    short sysvideo_28_16 ; not correct !
13839                              <1>
13840 0001059C 88DA                <1>       mov    dl, bl
13841                              <1>
13842                              <1>       ; ecx = cl = options
13843 0001059E E80E36FFFF          <1>       call   vbe_srs_gbs
13844                              <1>       ; ebx = state buffer size (data size)
13845                              <1>
13846 000105A3 891D[64030300]      <1>       mov    [u.r0], ebx
13847                              <1>
13848 000105A9 F6C240              <1>       test   dl, 64 ; verify without transfer
13849 000105AC 75AE                <1>       jnz    short sysvideo_28_16 ; yes
13850                              <1>
13851 000105AE BE00580900          <1>       mov    esi, VBE3VIDEOSTATE
13852 000105B3 BF00760900          <1>       mov    edi, VBE3SAVERESTOREBLOCK
13853 000105B8 87CB                <1>       xchg   ecx, ebx
13854 000105BA F3A4                <1>       rep    movsb
13855                              <1>
13856 000105BC 88D9                <1>       mov    cl, bl
13857                              <1>
13858                              <1>       ; 23/01/2021
13859 000105BE EB44                <1>       jmp    short sysvideo_28_10
13860                              <1>
13861                              <1> sysvideo_28_4:
13862 000105C0 53                  <1>       push   ebx
13863                              <1>       ; 24/01/2021
```

```
13864 000105C1 31DB              <1>       xor   ebx, ebx ; 0 ; use kernel's buffer
13865 000105C3 881D[69120300]    <1>       mov   [srvso], bl ; 0 ; invalidate
13866 000105C9 891D[6A120300]    <1>       mov   [VideoStateID], ebx ; 0 ; invalidate
13867 000105CF 0FB6CA            <1>       movzx ecx, dl ; options
13868 000105D2 B201              <1>       mov   dl, 1 ; save state
13869 000105D4 E890000000        <1>       call  sysvideo_28_11 ; 23/01/2021
13870                            <1>       ; Note: VBE3 BIOS data save option will be
13871                            <1>       ;     disabled.. ; 24/01/2021
13872 000105D9 89CA              <1>       mov   edx, ecx ; state (save) options
13873 000105DB 5B                <1>       pop   ebx
13874                            <1>
13875 000105DC 6683F84F          <1>       cmp   ax, 4Fh ; successful ?
13876 000105E0 7536              <1>       jne   short sysvideo_28_3 ; no !
13877                            <1>
13878 000105E2 F6C340            <1>       test  bl, 64 ; verify without transfer
13879 000105E5 7536              <1>       jnz   short sysvideo_28_6 ; yes
13880                            <1>
13881                            <1>       ; ecx = cl = options
13882 000105E7 E8C535FFFF        <1>       call  vbe_srs_gbs
13883                            <1>       ; ebx = state buffer size (data size)
13884                            <1>
13885 000105EC BE00760900        <1>       mov   esi, VBE3SAVERESTOREBLOCK
13886 000105F1 BF00580900        <1>       mov   edi, VBE3VIDEOSTATE
13887 000105F6 89D9              <1>       mov   ecx, ebx
13888 000105F8 F3A4              <1>       rep   movsb
13889                            <1>
13890 000105FA 88D1              <1>       mov   cl, dl
13891 000105FC 80C980            <1>       or    cl, 80h ; SVGA (VESA VBE) flag
13892                            <1>       ;mov  [srvso], dl
13893                            <1>
13894 000105FF E908010000        <1>       jmp   sysvideo_28_15
13895                            <1>
13896                            <1>       ; 23/01/2021
13897                            <1> sysvideo_28_10:
13898                            <1>       ; CL = VESA VBE3 Save/Restore options
13899                            <1>
13900 00010604 B202              <1>       mov   dl, 2 ; restore state
13901                            <1>
13902 00010606 E85C000000        <1>       call  sysvideo_28_1
13903                            <1>
13904 0001060B 6683F84F          <1>       cmp   ax, 4Fh ; successful ?
13905 0001060F 7407              <1>       je    short sysvideo_28_3
13906                            <1>       ;jmp  short sysvideo_28_9
13907                            <1>
13908                            <1> sysvideo_28_9:
13909                            <1>       ; return zero size (error) to user
13910 00010611 29C0              <1>       sub   eax, eax
13911                            <1> sysvideo_28_5:
13912 00010613 A3[64030300]      <1>       mov   [u.r0], eax
13913                            <1> sysvideo_28_3:
13914 00010618 E9B3CDFFFF        <1>       jmp   sysret
13915                            <1>
13916                            <1> sysvideo_28_6:
13917                            <1>       ; use timer ticks as VideoStateID
13918 0001061D A1[28810100]      <1>       mov   eax, [TIMER_LH]
13919 00010622 09C0              <1>       or    eax, eax
13920 00010624 75ED              <1>       jnz   short sysvideo_28_5
13921 00010626 40                <1>       inc   eax
13922 00010627 EBEA              <1>       jmp   short sysvideo_28_5
13923                            <1>
13924                            <1> sysvideo_28_7:
13925                            <1>       ; save/restore to/from user buffer
13926                            <1>
13927                            <1>       ; 23/01/2021
13928 00010629 89CE              <1>       mov   esi, ecx ; user's vstate buffer
13929 0001062B BF00760900        <1>       mov   edi, VBE3SAVERESTOREBLOCK
13930                            <1>
13931 00010630 0FB6CA            <1>       movzx ecx, dl ; VESA VBE func 4F04h options
13932                            <1>
13933                            <1>       ; source or destination is user buffer
13934 00010633 F6C301            <1>       test  bl, 1
13935 00010636 7444              <1>       jz    short sysvideo_28_12  ; Save
13936                            <1>
13937                            <1>       ; Restore
13938 00010638 803D[68120300]00  <1>       cmp   byte [srvsf], 0 ; srs permission flag
13939 0001063F 766A              <1>       jna   short sysvideo_28_14 ; not permitted
13940                            <1>
13941 00010641 88DA              <1>       mov   dl, bl ; 'sysvideo' options
13942                            <1>
13943                            <1>       ; ecx = cl = options
13944 00010643 E86935FFFF        <1>       call  vbe_srs_gbs
13945                            <1>       ; ebx = state buffer size (data size)
13946                            <1>
13947 00010648 891D[64030300]    <1>       mov   [u.r0], ebx ; transfer count
13948                            <1>
13949 0001064E F6C240            <1>       test  dl, 64 ; verify without transfer
13950 00010651 7558              <1>       jnz   short sysvideo_28_14 ; yes
13951                            <1>
13952 00010653 6681FB0008        <1>       cmp   bx, 2048
13953 00010658 73B7              <1>       jnb   short sysvideo_28_9 ; invalid
13954                            <1>
13955 0001065A 87CB              <1>       xchg  ecx, ebx
13956                            <1>       ; esi = user buffer
13957                            <1>       ; edi = VBE3SAVERESTOREBLOCK
13958                            <1>
13959 0001065C E82B0F0000        <1>       call  transfer_from_user_buffer
13960 00010661 72AE              <1>       jc    short sysvideo_28_9 ; error
13961                            <1>
13962 00010663 89D9              <1>       mov   ecx, ebx ; Function 4F04h options
13963 00010665 EB9D              <1>       jmp   short sysvideo_28_10 ; 23/01/2021
13964                            <1>
13965                            <1> sysvideo_28_1:
13966 00010667 31DB              <1>       xor   ebx, ebx ; 0 ; use kernel's buffer
13967                            <1> sysvideo_28_11:
13968                            <1>       ; 24/01/2021
```

```
13969 00010669 803D[4E090000]03    <1>       cmp    byte [vbe3], 3
13970 00010670 7405                <1>       je     short sysvideo_28_2
13971                              <1>
13972                              <1>       ; VESA VBE2 (BOCHS/QEMU/VBOX) video bios
13973 00010672 E9A334FFFF          <1>       jmp    _vbe_biosfn_save_restore_state
13974                              <1> sysvideo_28_2:
13975                              <1>       ;24/01/2021
13976                              <1>       ;mov   eax, 4F04h  ; Save/Restore vstate
13977                              <1>       ; VESA VBE3 video bios
13978 00010677 E94C13FFFF          <1>       jmp    _vbe3_pmfn_save_restore_state
13979                              <1>
13980                              <1> sysvideo_28_12:
13981                              <1>       ; Save
13982                              <1>       ;mov   edi, VBE3SAVERESTOREBLOCK
13983                              <1>
13984                              <1>       ;movzx ecx, dl ; options
13985 0001067C 56                  <1>       push   esi
13986 0001067D 53                  <1>       push   ebx
13987                              <1>       ; 23/01/2021
13988 0001067E B201                <1>       mov    dl, 1 ; save state
13989 00010680 E8E2FFFFFF          <1>       call   sysvideo_28_1
13990 00010685 5A                  <1>       pop    edx ; 'sysvideo' options
13991 00010686 5F                  <1>       pop    edi ; user's video state buffer
13992                              <1>
13993 00010687 6683F84F            <1>       cmp    ax, 4Fh ; successful ?
13994 0001068B 751E                <1>       jne    short sysvideo_28_14 ; no !
13995                              <1>
13996                              <1>       ; ecx = cl = options
13997 0001068D E81F35FFFF          <1>       call   vbe_srs_gbs
13998                              <1>       ; ebx = state buffer size (data size)
13999                              <1>
14000 00010692 89D9                <1>       mov    ecx, ebx ; transfer count
14001                              <1>
14002 00010694 F6C240              <1>       test   dl, 64 ; verify without transfer
14003 00010697 750C                <1>       jnz    short sysvideo_28_13 ; yes
14004                              <1>
14005                              <1>       ;mov   edi, esi
14006 00010699 BE00760900          <1>       mov    esi, VBE3SAVERESTOREBLOCK
14007 0001069E E89F0E0000          <1>       call   transfer_to_user_buffer
14008 000106A3 7206                <1>       jc     short sysvideo_28_14
14009                              <1> sysvideo_28_13:
14010 000106A5 890D[64030300]      <1>       mov    [u.r0], ecx
14011                              <1> sysvideo_28_14:
14012 000106AB E920CDFFFF          <1>       jmp    sysret
14013                              <1>
14014                              <1> sysvideo_29:
14015                              <1>       ; 18/01/2021
14016                              <1>       ; BH = 13
14017                              <1>       ; Save/Restore std VGA video state
14018                              <1>
14019                              <1>       ; bl = 0..3
14020                              <1>       ;      save to or restore from
14021                              <1>       ;      system buffer, VBE3VIDEOSTATE
14022                              <1>       ;      ECX = VideoStateID for restoring
14023                              <1>       ; bl = 4..7
14024                              <1>       ;      save to or restore from
14025                              <1>       ;      user buffer pointed by ECX
14026                              <1>
14027 000106B0 80FB03              <1>       cmp    bl, 3
14028 000106B3 7776                <1>       ja     short sysvideo_29_6
14029                              <1>
14030                              <1>       ; source or destination is kernel/system buffer
14031                              <1>
14032 000106B5 803D[68120300]00    <1>       cmp    byte [srvsf], 0 ; srs permission flag
14033 000106BC 7668                <1>       jna    short sysvideo_29_5 ; not permitted
14034                              <1>
14035 000106BE F6C301              <1>       test   bl, 1
14036 000106C1 7437                <1>       jz     short sysvideo_29_2  ; Save
14037                              <1>
14038                              <1>       ; Restore
14039 000106C3 3B0D[6A120300]      <1>       cmp    ecx, [VideoStateID]
14040 000106C9 755B                <1>       jne    short sysvideo_29_5 ; not correct !
14041 000106CB 80FB01              <1>       cmp    bl, 1
14042 000106CE 7709                <1>       ja     short sysvideo_29_0
14043                              <1>       ; bl = 1
14044 000106D0 BB6E000000          <1>       mov    ebx, 110
14045 000106D5 B103                <1>       mov    cl, 3 ; ctrl, vbios data
14046 000106D7 EB07                <1>       jmp    short sysvideo_29_1
14047                              <1> sysvideo_29_0:
14048                              <1>       ; bl  = 3
14049 000106D9 BB72030000          <1>       mov    ebx, 882
14050 000106DE B107                <1>       mov    cl, 7 ; ctrl, vbios data, dac
14051                              <1> sysvideo_29_1:
14052 000106E0 3A0D[69120300]      <1>       cmp    cl, [srvso]
14053 000106E6 753E                <1>       jne    short sysvideo_29_5 ; not correct !
14054                              <1>
14055 000106E8 BE00580900          <1>       mov    esi, VBE3VIDEOSTATE ; 22/01/2021
14056 000106ED E85936FFFF          <1>       call   biosfn_restore_video_state
14057 000106F2 891D[64030300]      <1>       mov    [u.r0], ebx ; video state size (bytes)
14058                              <1>       ;jmp   sysret
14059 000106F8 EB2C                <1>       jmp    short sysvideo_29_5
14060                              <1> sysvideo_29_2:
14061                              <1>       ;mov   esi, ecx
14062 000106FA BF00580900          <1>       mov    edi, VBE3VIDEOSTATE
14063                              <1>
14064 000106FF B107                <1>       mov    cl, 7 ; ctrl, vbios data, dac
14065 00010701 08DB                <1>       or     bl, bl
14066 00010703 7502                <1>       jnz    short sysvideo_29_3 ; bl = 2
14067                              <1>       ; bl = 0
14068 00010705 B103                <1>       mov    cl, 3 ; ctrl, vbios data
14069                              <1> sysvideo_29_3:
14070 00010707 E8D134FFFF          <1>       call   biosfn_save_video_state
14071                              <1> sysvideo_28_15:
14072                              <1>       ; use timer ticks as VideoStateID
14073 0001070C A1[28810100]        <1>       mov    eax, [TIMER_LH]
```

```
14074 00010711 21C0                <1>       and    eax, eax
14075 00010713 7501                <1>       jnz    short sysvideo_29_4
14076 00010715 40                  <1>       inc    eax
14077                              <1> sysvideo_29_4:
14078 00010716 880D[69120300]      <1>       mov    [srvso], cl
14079 0001071C A3[6A120300]        <1>       mov    [VideoStateID], eax
14080 00010721 A3[64030300]        <1>       mov    [u.r0], eax
14081                              <1> sysvideo_29_5:
14082 00010726 E9A5CCFFFF          <1>       jmp    sysret
14083                              <1>
14084                              <1> sysvideo_29_6:
14085 0001072B 80FB07              <1>       cmp    bl, 7
14086 0001072E 77F6                <1>       ja     short sysvideo_29_5 ; invalid sub function
14087                              <1>
14088 00010730 89CE                <1>       mov    esi, ecx
14089 00010732 BF00760900          <1>       mov    edi, VBE3SAVERESTOREBLOCK
14090                              <1>
14091                              <1>       ; source or destination is user buffer
14092 00010737 F6C301              <1>       test   bl, 1
14093 0001073A 7434                <1>       jz     short sysvideo_29_9  ; Save
14094                              <1>
14095                              <1>       ; Restore
14096 0001073C 803D[68120300]00    <1>       cmp    byte [srvsf], 0 ; srs permission flag
14097 00010743 76E1                <1>       jna    short sysvideo_29_5 ; not permitted
14098                              <1>
14099                              <1>       ;mov   esi, ecx
14100                              <1>       ;mov   edi, VBE3SAVERESTOREBLOCK
14101                              <1>
14102 00010745 80FB07              <1>       cmp    bl, 7
14103 00010748 7409                <1>       je     short sysvideo_29_7
14104                              <1>       ; bl = 5
14105 0001074A B303                <1>       mov    bl, 3
14106 0001074C B96E000000          <1>       mov    ecx, 110
14107 00010751 EB05                <1>       jmp    short sysvideo_29_8
14108                              <1> sysvideo_29_7:
14109                              <1>       ; bl = 7
14110 00010753 B972030000          <1>       mov    ecx, 882
14111                              <1> sysvideo_29_8:
14112 00010758 E82F0E0000          <1>       call   transfer_from_user_buffer
14113 0001075D 72C7                <1>       jc     short sysvideo_29_5
14114 0001075F 890D[64030300]      <1>       mov    [u.r0], ecx
14115 00010765 88D9                <1>       mov    cl, bl ; mov cl,7 (mov cl,3)
14116 00010767 89FE                <1>       mov    esi, edi ; VBE3SAVERESTOREBLOCK
14117                              <1>       ; cl = 3 or 7
14118 00010769 E8DD35FFFF          <1>       call   biosfn_restore_video_state
14119 0001076E EBB6                <1>       jmp    sysvideo_29_5
14120                              <1>       ;jmp   sysret
14121                              <1> sysvideo_29_9:
14122                              <1>       ; Save
14123                              <1>       ;mov   edi, VBE3SAVERESTOREBLOCK
14124                              <1>
14125 00010770 80FB06              <1>       cmp    bl, 6
14126 00010773 7409                <1>       je     short sysvideo_29_10
14127                              <1>       ; bl = 4
14128 00010775 BB6E000000          <1>       mov    ebx, 110
14129 0001077A B103                <1>       mov    cl, 3 ; ctrl, vbios data
14130 0001077C EB07                <1>       jmp    short sysvideo_29_11
14131                              <1> sysvideo_29_10:
14132                              <1>       ; bl = 6
14133 0001077E BB72030000          <1>       mov    ebx, 882
14134 00010783 B107                <1>       mov    cl, 7 ; ctrl, vbios data, dac
14135                              <1> sysvideo_29_11:
14136 00010785 E85334FFFF          <1>       call   biosfn_save_video_state
14137                              <1>
14138 0001078A 89D9                <1>       mov    ecx, ebx ; transfer count
14139 0001078C 89F7                <1>       mov    edi, esi
14140 0001078E BE00760900          <1>       mov    esi, VBE3SAVERESTOREBLOCK
14141                              <1>
14142                              <1>       ;call  transfer_to_user_buffer
14143                              <1>       ;jc    short sysvideo_29_5
14144                              <1>       ;mov   [u.r0], ecx ; transfer count
14145                              <1>       ;;jmp  sysret
14146                              <1>       ;jmp   short sysvideo_29_5
14147                              <1>
14148 00010793 EB1A                <1>       jmp    short sysvideo_29_12
14149                              <1>
14150                              <1> sysvideo_30:
14151 00010795 80FF0F              <1>       cmp    bh, 15
14152 00010798 7722                <1>       ja     short sysvideo_31 ; invalid function
14153                              <1>
14154                              <1>       ; BH = 15
14155                              <1>       ; Copy VESA EDID to user's buffer
14156                              <1>
14157 0001079A 803D[6D420000]4F    <1>       cmp    byte [edid], 4Fh
14158 000107A1 7519                <1>       jne    short sysvideo_31 ; not ready !
14159                              <1>
14160                              <1>       ;and   ecx, ecx
14161                              <1>       ;jz    short sysvideo_31
14162                              <1>
14163                              <1>       ; ecx = user's buffer address
14164 000107A3 89CF                <1>       mov    edi, ecx
14165 000107A5 BE[84110300]        <1>       mov    esi, edid_info
14166 000107AA B980000000          <1>       mov    ecx, 128 ; 128 bytes
14167                              <1> sysvideo_29_12:
14168 000107AF E88E0D0000          <1>       call   transfer_to_user_buffer
14169 000107B4 7206                <1>       jc     short sysvideo_31
14170                              <1>
14171 000107B6 890D[64030300]      <1>       mov    [u.r0], ecx ; EDID size, 128 bytes
14172                              <1> sysvideo_31:
14173 000107BC E90FCCFFFF          <1>       jmp    sysret
14174                              <1>
14175                              <1> mkdir:
14176                              <1>       ; 04/12/2015 (14 byte directory names)
14177                              <1>       ; 12/10/2015
14178                              <1>       ; 17/06/2015 (Retro UNIX 386 v1 - Beginning)
```

```
14179                                  <1>       ; 29/04/2013 - 01/08/2013 (Retro UNIX 8086 v1)
14180                                  <1>       ;
14181                                  <1>       ; 'mkdir' makes a directory entry from the name pointed to
14182                                  <1>       ; by u.namep into the current directory.
14183                                  <1>       ;
14184                                  <1>       ; INPUTS ->
14185                                  <1>       ;    u.namep - points to a file name
14186                                  <1>       ;                  that is about to be a directory entry.
14187                                  <1>       ;    ii - current directory's i-number.
14188                                  <1>       ; OUTPUTS ->
14189                                  <1>       ;    u.dirbuf+2 - u.dirbuf+10 - contains file name.
14190                                  <1>       ;    u.off - points to entry to be filled
14191                                  <1>       ;           in the current directory
14192                                  <1>       ;    u.base - points to start of u.dirbuf.
14193                                  <1>       ;    r1 - contains i-number of current directory
14194                                  <1>       ;
14195                                  <1>       ; ((AX = R1)) output
14196                                  <1>       ;
14197                                  <1>       ;    (Retro UNIX Prototype : 11/11/2012, UNIXCOPY.ASM)
14198                                  <1>       ;    ((Modified registers: eAX, eDX, eBX, eCX, eSI, eDI, eBP))
14199                                  <1>       ;
14200                                  <1>
14201                                  <1>       ; 17/06/2015 - 32 bit modifications (Retro UNIX 386 v1)
14202 000107C1 31C0                   <1>       xor    eax, eax
14203 000107C3 BF[9A030300]           <1>       mov     edi, u.dirbuf+2
14204 000107C8 89FE                   <1>       mov    esi, edi
14205 000107CA AB                     <1>       stosd
14206 000107CB AB                     <1>       stosd
14207                                  <1>       ; 04/12/2015 (14 byte directory names)
14208 000107CC AB                     <1>       stosd
14209 000107CD 66AB                   <1>       stosw
14210                                  <1>             ; jsr r0,copyz; u.dirbuf+2; u.dirbuf+10. / clear this
14211 000107CF 89F7                   <1>       mov    edi, esi ; offset to u.dirbuf
14212                                  <1>       ; 12/10/2015 ([u.namep] -> ebp)
14213                                  <1>       ;mov   ebp, [u.namep]
14214 000107D1 E8F6020000             <1>       call   trans_addr_nmbp ; convert virtual address to physical
14215                                  <1>             ; esi = physical address (page start + offset)
14216                                  <1>             ; ecx = byte count in the page (1 - 4096)
14217                                  <1>       ; edi = offset to u.dirbuf (edi is not modified in trans_addr_nm)
14218                                  <1>             ; mov u.namep,r2 / r2 points to name of directory entry
14219                                  <1>             ; mov $u.dirbuf+2,r3 / r3 points to u.dirbuf+2
14220                                  <1> mkdir_1: ; 1:
14221 000107D6 45                     <1>       inc    ebp ; 12/10/2015
14222                                  <1>       ;
14223                                  <1>       ; / put characters in the directory name in u.dirbuf+2 - u.dirbuf+10
14224                                  <1>        ; 01/08/2013
14225 000107D7 AC                     <1>       lodsb
14226                                  <1>             ; movb (r2)+,r1 / move character in name to r1
14227 000107D8 20C0                   <1>       and    al, al
14228 000107DA 7427                   <1>       jz     short mkdir_3
14229                                  <1>             ; beq 1f / if null, done
14230 000107DC 3C2F                   <1>       cmp    al, '/'
14231                                  <1>             ; cmp r1,$'/ / is it a "/"?
14232 000107DE 7414                   <1>       je     short mkdir_err
14233                                  <1>       ;je    error
14234                                  <1>             ; beq error9 / yes, error
14235                                  <1>       ; 12/10/2015
14236 000107E0 6649                   <1>       dec    cx
14237 000107E2 7505                   <1>       jnz    short mkdir_2
14238                                  <1>       ; 12/10/2015 ([u.namep] -> ebp)
14239 000107E4 E8E9020000             <1>       call   trans_addr_nm ; convert virtual address to physical
14240                                  <1>             ; esi = physical address (page start + offset)
14241                                  <1>             ; ecx = byte count in the page
14242                                  <1>       ; edi = offset to u.dirbuf (edi is not modified in trans_addr_nm)
14243                                  <1> mkdir_2:
14244 000107E9 81FF[A8030300]         <1>       cmp     edi, u.dirbuf+16 ; ; 04/12/2015 (10 -> 16)
14245                                  <1>             ; cmp r3,$u.dirbuf+10. / have we reached the last slot for
14246                                  <1>                                    ; / a char?
14247 000107EF 74E5                   <1>       je     short mkdir_1
14248                                  <1>             ; beq 1b / yes, go back
14249 000107F1 AA                     <1>       stosb
14250                                  <1>             ; movb r1,(r3)+ / no, put the char in the u.dirbuf
14251 000107F2 EBE2                   <1>       jmp    short mkdir_1
14252                                  <1>             ; br 1b / get next char
14253                                  <1> mkdir_err:
14254                                  <1>       ; 17/06/2015
14255 000107F4 C705[C8030300]1300-    <1>       mov    dword [u.error], ERR_NOT_DIR ; 'not a valid directory !'
14255 000107FC 0000                   <1>
14256 000107FE E9ADCBFFFF             <1>       jmp    error
14257                                  <1>
14258                                  <1> mkdir_3: ; 1:
14259 00010803 A1[78030300]           <1>       mov    eax, [u.dirp]
14260 00010808 A3[80030300]           <1>       mov    [u.off], eax
14261                                  <1>             ; mov u.dirp,u.off / pointer to empty current directory
14262                                  <1>                                ; / slot to u.off
14263                                  <1> wdir: ; 29/04/2013
14264 0001080D C705[84030300]-        <1>       mov   dword [u.base], u.dirbuf
14264 00010813 [98030300]             <1>
14265                                  <1>             ; mov $u.dirbuf,u.base / u.base points to created file name
14266 00010817 C705[88030300]1000-    <1>       mov   dword [u.count], 16 ; 04/12/2015 (10 -> 16)
14266 0001081F 0000                   <1>
14267                                  <1>             ; mov $10.,u.count / u.count = 10
14268 00010821 66A1[51040300]         <1>       mov    ax, [ii]
14269                                  <1>             ; mov ii,r1 / r1 has i-number of current directory
14270 00010827 B201                   <1>       mov    dl, 1 ; owner flag mask ; RETRO UNIX 8086 v1 modification !
14271 00010829 E85D1D0000             <1>       call   access
14272                                  <1>             ; jsr r0,access; 1 / get i-node and set its file up
14273                                  <1>                                ; / for writing
14274                                  <1>       ; AX = i-number of current directory
14275                                  <1>       ; 01/08/2013
14276 0001082E FE05[C6030300]         <1>       inc     byte [u.kcall] ; the caller is 'mkdir' sign
14277 00010834 E86B0F0000             <1>       call   writei
14278                                  <1>             ; jsr r0,writei / write into directory
14279 00010839 C3                     <1>       retn
14280                                  <1>             ; rts r0
```

```
14281                                 <1>
14282                                 <1> sysexec:
14283                                 <1>       ; 18/11/2017
14284                                 <1>       ; 14/11/2017
14285                                 <1>       ; 13/11/2017
14286                                 <1>       ; 24/10/2016, 04/01/2017
14287                                 <1>       ; 24/04/2016 - TRDOS 386 (TRDOS v2.0)
14288                                 <1>       ; 23/06/2015 - 23/10/2015 (Retro UNIX 386 v1)
14289                                 <1>       ; 03/06/2013 - 06/12/2013 (Retro UNIX 8086 v1)
14290                                 <1>       ;
14291                                 <1>       ; 'sysexec' initiates execution of a file whose path name if
14292                                 <1>       ; pointed to by 'name' in the sysexec call.
14293                                 <1>       ; 'sysexec' performs the following operations:
14294                                 <1>       ;    1. obtains i-number of file to be executed via 'namei'.
14295                                 <1>       ;    2. obtains i-node of file to be exceuted via 'iget'.
14296                                 <1>       ;    3. sets trap vectors to system routines.
14297                                 <1>       ;    4. loads arguments to be passed to executing file into
14298                                 <1>       ;      highest locations of user's core
14299                                 <1>       ;    5. puts pointers to arguments in locations immediately
14300                                 <1>       ;      following arguments.
14301                                 <1>       ;    6.       saves number of arguments in next location.
14302                                 <1>       ;    7. intializes user's stack area so that all registers
14303                                 <1>       ;      will be zeroed and the PS is cleared and the PC set
14304                                 <1>       ;      to core when 'sysret' restores registers
14305                                 <1>       ;      and does an rti.
14306                                 <1>       ;    8. inializes u.r0 and u.sp
14307                                 <1>       ;    9. zeros user's core down to u.r0
14308                                 <1>       ;   10.       reads executable file from storage device into core
14309                                 <1>       ;      starting at location 'core'.
14310                                 <1>       ;   11.       sets u.break to point to end of user's code with
14311                                 <1>       ;      data area appended.
14312                                 <1>       ;   12.       calls 'sysret' which returns control at location
14313                                 <1>       ;      'core' via 'rti' instruction.
14314                                 <1>       ;
14315                                 <1>       ; Calling sequence:
14316                                 <1>       ;     sysexec; namep; argp
14317                                 <1>       ; Arguments:
14318                                 <1>       ;     namep - points to pathname of file to be executed
14319                                 <1>       ;     argp  - address of table of argument pointers
14320                                 <1>       ;     argp1... argpn - table of argument pointers
14321                                 <1>       ;     argp1:<...0> ... argpn:<...0> - argument strings
14322                                 <1>       ; Inputs: (arguments)
14323                                 <1>       ; Outputs: -
14324                                 <1>       ; ..........................................................
14325                                 <1>       ;
14326                                 <1>       ; Retro UNIX 386 v1 modification:
14327                                 <1>       ;     User application runs in it's own virtual space
14328                                 <1>       ;     which is izolated from kernel memory (and other
14329                                 <1>       ;     memory pages) via 80386   paging in ring 3
14330                                 <1>       ;     privilige mode. Virtual start address is always 0.
14331                                 <1>       ;     User's core memory starts at linear address 400000h
14332                                 <1>       ;     (the end of the 1st 4MB).
14333                                 <1>       ;
14334                                 <1>       ; Retro UNIX 8086 v1 modification:
14335                                 <1>       ;     user/application segment and system/kernel segment
14336                                 <1>       ;     are different and sysenter/sysret/sysrele routines
14337                                 <1>       ;     are different (user's registers are saved to
14338                                 <1>       ;     and then restored from system's stack.)
14339                                 <1>       ;
14340                                 <1>       ;     NOTE: Retro UNIX 8086 v1 'arg2' routine gets these
14341                                 <1>       ;           arguments which were in these registers;
14342                                 <1>       ;           but, it returns by putting the 1st argument
14343                                 <1>       ;           in 'u.namep' and the 2nd argument
14344                                 <1>       ;           on top of stack. (1st argument is offset of the
14345                                 <1>       ;           file/path name in the user's program segment.)
14346                                 <1>
14347                                 <1>       ;call  arg2
14348                                 <1>       ; * name - 'u.namep' points to address of file/path name
14349                                 <1>       ;          in the user's program segment ('u.segmnt')
14350                                 <1>       ;          with offset in BX register (as sysopen argument 1).
14351                                 <1>       ; * argp - sysexec argument 2 is in CX register
14352                                 <1>       ;          which is on top of stack.
14353                                 <1>       ;
14354                                 <1>       ;       ; jsr r0,arg2 / arg0 in u.namep,arg1 on top of stack
14355                                 <1>
14356                                 <1>       ; 23/06/2015 (32 bit modifications)
14357                                 <1>
14358                                 <1>       ;; 13/11/2017
14359                                 <1>       ;;mov  [u.namep], ebx ; argument 1
14360                                 <1>         ; 18/10/2015
14361 0001083A 890D[4C040300]        <1>       mov     [argv], ecx  ; * ; argument 2
14362                                 <1>
14363                                 <1>       ; 13/11/2017
14364 00010840 89DE                  <1>       mov    esi, ebx
14365 00010842 E877210000            <1>       call   set_working_path_x
14366 00010847 7319                  <1>       jnc    short sysexec_0
14367                                 <1>
14368                                 <1>       ;; 'bad command or file name'
14369                                 <1>       ;mov   eax, ERR_BAD_CMD_ARG ; 01h ; TRDOS 8086
14370                                 <1>
14371                                 <1>       ; 'file not found !' error
14372 00010849 B802000000            <1>       mov    eax, ERR_NOT_FOUND ; 02h ; TRDOS 8086
14373                                 <1> sysexec_not_found_err:
14374                                 <1> sysexec_access_error:
14375                                 <1> sysexec_ext_error:
14376 0001084E A3[64030300]          <1>       mov    [u.r0], eax
14377 00010853 A3[C8030300]          <1>       mov    [u.error], eax
14378 00010858 E836220000            <1>       call   reset_working_path
14379 0001085D E94ECBFFFF            <1>       jmp    error
14380                                 <1>
14381                                 <1> sysexec_0:
14382                                 <1>       ; 13/11/2017
14383                                 <1>       ;mov   esi, FindFile_Name
14384 00010862 66B80018              <1>         mov  ax, 1800h ; Only files
14385 00010866 E8EA86FFFF            <1>       call   find_first_file
```

```
14386 0001086B 72E1                <1>       jc     short sysexec_not_found_err ; eax = 2
14387                              <1>
14388                              <1>       ; check_ file attributes
14389                              <1>       ; (attribute bits = 00ADVSHR) ; 18h = Directory+Volume
14390                              <1>       ; BL = Attributes byte
14391                              <1>
14392 0001086D F6C306             <1>       test   bl, 6  ; system file or hidden file (S+H)
14393                              <1>       ;jz    short sysexec_0ext
14394 00010870 7417                <1>       jz     short sysexec_1 ; yes
14395                              <1>
14396                              <1>       ; 13/11/2017
14397                              <1>       ; /// TRDOS386 permission check for multiuser mode ///
14398                              <1>       ; SYSTEM file or HIDDEN file !!
14399                              <1>       ; (Only super user has permission to run this file.)
14400                              <1>
14401                              <1>       ; ([u.uid]=0 for super user or root in multiuser mode)
14402                              <1>       ; ([u.uid]=0 for any users in singleuser mode)
14403 00010872 803D[B0030300]00    <1>       cmp    byte [u.uid], 0 ; Super User ([u.uid]=0) ?
14404                              <1>       ;jna   short sysexec_0ext
14405 00010879 760E                <1>       jna    short sysexec_1 ; yes
14406                              <1>
14407                              <1>       ; 'permission denied !' error
14408 0001087B B80B000000          <1>       mov    eax, ERR_FILE_ACCESS  ; 11 = ERR_PERM_DENIED
14409 00010880 EBCC                <1>       jmp    short sysexec_access_error
14410                              <1>
14411                              <1> sysexec_not_exf:
14412                              <1>       ; 'not executable file !' error
14413 00010882 B816000000          <1>       mov    eax, ERR_NOT_EXECUTABLE
14414 00010887 EBC5                <1>       jmp    sysexec_ext_error
14415                              <1>
14416                              <1> ;sysexec_0ext:
14417                              <1> sysexec_1:
14418                              <1>       ; 18/11/2017
14419 00010889 BE[508A0100]        <1>       mov    esi, FindFile_Name
14420                              <1>       ; 13/11/2017
14421                              <1>       ; check program file name extension
14422                              <1>       ; ('.PRG' for current TRDOS version)
14423 0001088E E865A1FFFF          <1>       call   check_prg_filename_ext
14424 00010893 72ED                <1>       jc     short sysexec_not_exf
14425                              <1>
14426                              <1>       ; 18/11/2017
14427 00010895 3C50                <1>       cmp    al, 'P'
14428 00010897 75E9                <1>       jne    short sysexec_not_exf
14429                              <1>
14430                              <1>       ; '.PRG' extension is OK.
14431                              <1>       ; Only '.PRG' files are valid program files
14432                              <1>       ; for current TRDOS 386 version.
14433                              <1>
14434 00010899 8B15[7C8A0100]      <1>       mov    edx, [FindFile_DirEntry+DirEntry_FileSize]
14435 0001089F 66A1[748A0100]      <1>       mov    ax, [FindFile_DirEntry+DirEntry_FstClusHI]
14436 000108A5 C1E010              <1>       shl    eax, 16
14437 000108A8 66A1[7A8A0100]      <1>       mov    ax, [FindFile_DirEntry+DirEntry_FstClusLO]
14438                              <1>       ; EAX = First Cluster number
14439                              <1>       ; EDX = File Size
14440                              <1>
14441 000108AE A3[51040300]        <1>       mov    [ii], eax
14442 000108B3 8915[55040300]      <1>       mov    [i.size], edx
14443                              <1>
14444                              <1> ;sysexec_1:
14445                              <1>       ; 13/11/2017 - TRDOS 386 (TRDOS v2.0)
14446                              <1>       ; 24/06/2015 - 23/10/2015 (Retro UNIX 386 v1)
14447                              <1>         ; Moving arguments to the end of [u.upage]
14448                              <1>       ; (by regarding page borders in user's memory space)
14449                              <1>       ;
14450                              <1>       ; 10/10/2015
14451                              <1>       ; 21/07/2015
14452 000108B9 89E5                <1>       mov    ebp, esp ; (**)
14453                              <1>       ; 18/10/2015
14454 000108BB 89EF                <1>       mov    edi, ebp
14455 000108BD B900010000          <1>       mov    ecx, MAX_ARG_LEN ; 256
14456                              <1>       ;sub   edi, MAX_ARG_LEN ; 256
14457 000108C2 29CF                <1>       sub    edi, ecx
14458 000108C4 89FC                <1>       mov    esp, edi ; *!*
14459 000108C6 31C0                <1>       xor    eax, eax
14460 000108C8 A3[8C030300]        <1>       mov    [u.nread], eax ; 0
14461 000108CD 66A3[4A040300]      <1>       mov    [argc], ax ; 0 ; 13/11/2017
14462 000108D3 49                  <1>       dec    ecx ; 256 - 1
14463 000108D4 890D[88030300]      <1>       mov    [u.count], ecx ; MAX_ARG_LEN - 1 ; 255
14464                              <1>       ;mov   dword [u.count], MAX_ARG_LEN - 1 ; 255
14465                              <1> sysexec_2:
14466 000108DA 8B35[4C040300]      <1>       mov    esi, [argv] ; 18/10/2015
14467 000108E0 E866000000          <1>       call   get_argp
14468 000108E5 B904000000          <1>       mov    ecx, 4 ; mov ecx, 4
14469                              <1> sysexec_3:
14470 000108EA 21C0                <1>       and    eax, eax
14471 000108EC 0F8458050000        <1>         jz     sysexec_6
14472                              <1>       ; 18/10/2015
14473 000108F2 010D[4C040300]      <1>       add    [argv], ecx ; 4
14474 000108F8 66FF05[4A040300]    <1>       inc    word [argc]
14475                              <1>       ;
14476 000108FF A3[84030300]        <1>       mov    [u.base], eax
14477                              <1>       ; 23/10/2015
14478 00010904 66C705[C4030300]00- <1>       mov    word [u.pcount], 0
14478 0001090C 00                  <1>
14479                              <1> sysexec_4:
14480 0001090D E8D00B0000          <1>       call   cpass ; get a character from user's core memory
14481 00010912 750E                <1>         jnz  short sysexec_5
14482                              <1>              ; (max. 255 chars + null)
14483                              <1>       ; 18/10/2015
14484 00010914 28C0                <1>       sub    al, al
14485 00010916 AA                  <1>       stosb
14486 00010917 FF05[8C030300]      <1>       inc    dword [u.nread]
14487 0001091D E928050000          <1>       jmp    sysexec_6 ; 24/04/2016
14488                              <1> sysexec_5:
14489 00010922 AA                  <1>       stosb
```

```
14490 00010923 20C0                <1>       and    al, al
14491 00010925 75E6                <1>       jnz    short sysexec_4
14492 00010927 B904000000          <1>       mov    ecx, 4
14493 0001092C 390D[48040300]      <1>       cmp    [ncount], ecx ; 4
14494 00010932 72A6                <1>       jb     short sysexec_2
14495 00010934 8B35[44040300]      <1>       mov    esi, [nbase]
14496 0001093A 010D[44040300]      <1>       add    [nbase], ecx ; 4
14497 00010940 66290D[48040300]    <1>       sub    [ncount], cx
14498 00010947 8B06                <1>       mov    eax, [esi]
14499 00010949 EB9F                <1>       jmp    short sysexec_3
14500                              <1>
14501                              <1> get_argp:
14502                              <1>       ; 14/11/2017 - TRDOS 386 (TRDOS v2.0)
14503                              <1>       ; 18/10/2015 (nbase, ncount)
14504                              <1>       ; 21/07/2015
14505                              <1>       ; 24/06/2015 (Retro UNIX 386 v1)
14506                              <1>       ; Get (virtual) address of argument from user's core memory
14507                              <1>       ;
14508                              <1>       ; INPUT:
14509                              <1>       ;      esi = virtual address of argument pointer
14510                              <1>       ; OUTPUT:
14511                              <1>       ;      eax = virtual address of argument
14512                              <1>       ;
14513                              <1>       ; Modified registers: EAX, EBX, ECX, EDX, ESI
14514                              <1>       ;
14515 0001094B 833D[BC030300]00    <1>       cmp    dword [u.ppgdir], 0 ; /etc/init ?
14516                              <1>                                  ; (the caller is kernel)
14517 00010952 7667                <1>       jna    short get_argpk
14518                              <1>       ;
14519 00010954 89F3                <1>       mov    ebx, esi
14520 00010956 E80755FFFF          <1>       call   get_physical_addr ; get physical address
14521 0001095B 0F8289000000        <1>       jc     get_argp_err
14522 00010961 A3[44040300]        <1>       mov    [nbase], eax ; physical address
14523 00010966 66890D[48040300]    <1>       mov    [ncount], cx ; remain byte count in page (1-4096)
14524 0001096D B804000000          <1>       mov    eax, 4 ; 21/07/2015
14525 00010972 6639C1              <1>       cmp    cx, ax ; 4
14526 00010975 735D                <1>       jnb    short get_argp2
14527 00010977 89F3                <1>       mov    ebx, esi
14528 00010979 01CB                <1>       add    ebx, ecx
14529 0001097B E8E254FFFF          <1>       call   get_physical_addr ; get physical address
14530 00010980 7268                <1>       jc     short get_argp_err
14531                              <1>       ;push  esi
14532 00010982 89C6                <1>       mov    esi, eax
14533 00010984 66870D[48040300]    <1>       xchg   cx, [ncount]
14534 0001098B 8735[44040300]      <1>       xchg   esi, [nbase]
14535 00010991 B504                <1>       mov    ch, 4
14536 00010993 28CD                <1>       sub    ch, cl
14537                              <1> get_argp0:
14538 00010995 AC                  <1>       lodsb
14539 00010996 6650                <1>       push   ax
14540 00010998 FEC9                <1>       dec    cl
14541 0001099A 75F9                <1>       jnz    short get_argp0
14542 0001099C 8B35[44040300]      <1>       mov    esi, [nbase]
14543                              <1>       ; 21/07/2015
14544 000109A2 0FB6C5              <1>       movzx  eax, ch
14545 000109A5 0105[44040300]      <1>       add    [nbase], eax
14546 000109AB 662905[48040300]    <1>       sub    [ncount], ax
14547                              <1> get_argp1:
14548 000109B2 AC                  <1>       lodsb
14549 000109B3 FECD                <1>       dec    ch
14550 000109B5 7447                <1>       jz     short get_argp3
14551 000109B7 6650                <1>       push   ax
14552 000109B9 EBF7                <1>       jmp    short get_argp1
14553                              <1> get_argpk:
14554                              <1>       ; Argument is in kernel's memory space
14555 000109BB 66C705[48040300]00- <1>       mov    word [ncount], PAGE_SIZE ; 4096
14555 000109C3 10                  <1>
14556 000109C4 8935[44040300]      <1>       mov    [nbase], esi
14557 000109CA 8305[44040300]04    <1>       add    dword [nbase], 4
14558 000109D1 8B06                <1>       mov    eax, [esi] ; virtual addr. = physical addr.
14559 000109D3 C3                  <1>       retn
14560                              <1> get_argp2:
14561                              <1>       ; 21/07/2015
14562                              <1>       ;mov   eax, 4
14563 000109D4 8B15[44040300]      <1>       mov    edx, [nbase] ; 18/10/2015
14564 000109DA 0105[44040300]      <1>       add    [nbase], eax
14565 000109E0 662905[48040300]    <1>       sub    [ncount], ax
14566                              <1>       ;
14567 000109E7 8B02                <1>       mov    eax, [edx]
14568 000109E9 C3                  <1>       retn
14569                              <1> get_argp_err:
14570 000109EA A3[C8030300]        <1>       mov    [u.error], eax
14571                              <1>       ; 14/11/2017
14572 000109EF B801000000          <1>       mov    eax, ERR_BAD_CMD_ARG ; 01h ; TRDOS 8086
14573 000109F4 A3[64030300]        <1>       mov    [u.r0], eax
14574 000109F9 E9B2C9FFFF          <1>       jmp    error
14575                              <1> get_argp3:
14576 000109FE B103                <1>       mov    cl, 3
14577                              <1> get_argp4:
14578 00010A00 C1E008              <1>       shl    eax, 8
14579 00010A03 665A                <1>       pop    dx
14580 00010A05 88D0                <1>       mov    al, dl
14581 00010A07 E2F7                <1>       loop   get_argp4
14582                              <1>       ;pop   esi
14583 00010A09 C3                  <1>       retn
14584                              <1>
14585                              <1> sysstat:
14586                              <1>       ; 13/01/2017 - TRDOS 386 (TRDOS v2.0)
14587                              <1>       ; temporary !
14588 00010A0A B801000000          <1>       mov    eax, ERR_INV_FNUMBER ; 'invalid function number !'
14589 00010A0F A3[C8030300]        <1>       mov    [u.error], eax
14590 00010A14 A3[64030300]        <1>       mov    [u.r0], eax
14591 00010A19 E992C9FFFF          <1>       jmp    error
14592                              <1>
14593                              <1> sysfstat:
```

```
14594                                  <1>       ; 13/01/2017 - TRDOS 386 (TRDOS v2.0)
14595                                  <1>       ; temporary !
14596 00010A1E B801000000              <1>       mov    eax, ERR_INV_FNUMBER ; 'invalid function number !'
14597 00010A23 A3[C8030300]            <1>       mov    [u.error], eax
14598 00010A28 A3[64030300]            <1>       mov    [u.r0], eax
14599 00010A2D E97EC9FFFF              <1>       jmp    error
14600                                  <1>
14601                                  <1> fclose:
14602                                  <1>       ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
14603                                  <1>       ;
14604                                  <1>       ; 18/06/2015 (Retro UNIX 386 v1 - Beginning)
14605                                  <1>       ;            (32 bit offset pointer modification)
14606                                  <1>       ; 19/04/2013 - 12/01/2014 (Retro UNIX 8086 v1)
14607                                  <1>       ;
14608                                  <1>       ; Given the file descriptor (index to the u.fp list)
14609                                  <1>       ; 'fclose' first gets the i-number of the file via 'getf'.
14610                                  <1>       ; If i-node is active (i-number > 0) the entry in
14611                                  <1>       ; u.fp list is cleared. If all the processes that opened
14612                                  <1>       ; that file close it, then fsp etry is freed and the file
14613                                  <1>       ; is closed. If not a return is taken.
14614                                  <1>       ; If the file has been deleted while open, 'anyi' is called
14615                                  <1>       ; to see anyone else has it open, i.e., see if it is appears
14616                                  <1>       ; in another entry in the fsp table. Upon return from 'anyi'
14617                                  <1>       ; a check is made to see if the file is special.
14618                                  <1>       ;
14619                                  <1>       ; INPUTS ->
14620                                  <1>       ;    r1 - contains the file descriptor (value=0,1,2...)
14621                                  <1>       ;    u.fp - list of entries in the fsp table
14622                                  <1>       ;    fsp - table of entries (4 words/entry) of open files.
14623                                  <1>       ; OUTPUTS ->
14624                                  <1>       ;    r1 - contains the same file descriptor
14625                                  <1>       ;    r2 - contains i-number
14626                                  <1>       ;
14627                                  <1>       ; ((AX = R1))
14628                                  <1>       ; ((Modified registers: eDX, eBX, eCX, eSI, eDI, eBP))
14629                                  <1>       ;
14630                                  <1>       ; Retro UNIX 8086 v1 modification : CF = 1
14631                                  <1>       ;              if i-number of the file is 0. (error)
14632                                  <1>       ;
14633                                  <1>       ; TRDOS 386 (06/10/2016)
14634                                  <1>       ;
14635                                  <1>       ; INPUT:
14636                                  <1>       ;     EAX = File Handle (File Descriptor, File Index)
14637                                  <1>       ;
14638                                  <1>       ; OUTPUT:
14639                                  <1>       ;     CF = 1 -> File not open !
14640                                  <1>       ;     CF = 0 -> OK!
14641                                  <1>       ;          EBX = File Number (System)
14642                                  <1>       ;          [cdev] = Logical DOS Drive Number
14643                                  <1>       ;          EAX = File Handle/Number (user)
14644                                  <1>       ;
14645                                  <1>       ; Modified Registers: EBX
14646                                  <1>
14647 00010A32 50                      <1>       push   eax ; File handle
14648                                  <1>
14649 00010A33 E848000000              <1>       call   getf
14650                                  <1>       ;jc    device_close ; eax = device number
14651                                  <1>       ; 17/04/2021 (temporary)
14652 00010A38 7306                    <1>       jnc    short _fclose_0
14653 00010A3A 58                      <1>       pop    eax
14654 00010A3B E9F1D0FFFF              <1>       jmp    rw2 ; file not open !
14655                                  <1> _fclose_0:
14656 00010A40 80BB[9A8D0100]01        <1>       cmp    byte [ebx+OF_MODE], 1 ; open mode ; 0 = empty entry
14657 00010A47 722E                    <1>       jb     short fclose_1            ; 1 = read, 2 = write
14658                                  <1>
14659 00010A49 83F801                  <1>       cmp    eax, 1 ; is the first cluster number > 0
14660 00010A4C 7229                    <1>       jb     short fclose_1 ; no, this is empty entry
14661                                  <1>
14662                                  <1> fclose_0:
14663 00010A4E FE8B[AE8D0100]          <1>       dec    byte [ebx+OF_OPENCOUNT] ; decrement the number of processes
14664                                  <1>                                    ; that have opened the file
14665 00010A54 7921                    <1>       jns    short fclose_1 ; jump if not negative (jump if bit 7 is 0)
14666                                  <1>                   ; if all processes haven't closed the file, return
14667                                  <1>       ;
14668                                  <1>       ; eax ; First cluster
14669 00010A56 31C0                    <1>       xor    eax, eax ; 0
14670 00010A58 8883[9A8D0100]          <1>       mov    [ebx+OF_MODE], al ; 0 = empty entry
14671                                  <1>       ;mov   [ebx+OF_STATUS], al ; 0 = empty entry
14672 00010A5E 66C1E302                <1>       shl    bx, 2
14673 00010A62 8983[688D0100]          <1>       mov    [ebx+OF_FCLUSTER], eax ; 0
14674 00010A68 8983[808E0100]          <1>       mov    [ebx+OF_CCLUSTER], eax ; 0
14675                                  <1>       ;mov   [ebx+OF_CCINDEX], eax ; 0
14676 00010A6E A3[74030300]            <1>       mov    [u.fofp], eax ; 0
14677 00010A73 66C1EB02                <1>       shr    bx, 2
14678                                  <1> fclose_1: ; 1:
14679 00010A77 58                      <1>       pop    eax ; File handle (File Descriptor, File Index)
14680 00010A78 C680[6A030300]00        <1>       mov    byte [eax+u.fp], 0 ; clear that entry in the u.fp list
14681 00010A7F C3                      <1>       retn
14682                                  <1>
14683                                  <1> getf:
14684                                  <1>       ; 17/04/2021 - TRDOS 386 v2.0.4
14685                                  <1>       ;      (temporary modifications)
14686                                  <1>       ; 12/10/2016
14687                                  <1>       ; 11/10/2016
14688                                  <1>       ; 08/10/2016
14689                                  <1>       ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
14690                                  <1>       ; / get the device number and the i-number of an open file
14691                                  <1>       ; 13/05/2015
14692                                  <1>       ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
14693                                  <1>       ; 19/04/2013 - 18/11/2013 (Retro UNIX 8086 v1)
14694                                  <1>       ;
14695 00010A80 89C3                    <1>       mov    ebx, eax
14696                                  <1> getf1:
14697 00010A82 83FB0A                  <1>       cmp    ebx, 10
14698 00010A85 730A                    <1>         jnb  short getf2
```

```
14699 00010A87 8A9B[6A030300]      <1>       mov    bl, [ebx+u.fp]
14700 00010A8D 08DB                <1>       or     bl, bl
14701 00010A8F 7503                <1>       jnz    short getf3
14702                              <1> getf2:
14703                              <1>       ; 'File not open !' error (ax=0)
14704 00010A91 29C0                <1>       sub    eax, eax
14705 00010A93 C3                  <1>       retn
14706                              <1> getf3:
14707 00010A94 F6C380              <1>       test   bl, 80h
14708 00010A97 7531                <1>       jnz    short getf5 ; device
14709 00010A99 FECB                <1>       dec    bl ; 0 based
14710 00010A9B 8A83[908D0100]      <1>       mov    al, [ebx+OF_DRIVE]
14711 00010AA1 A2[46030300]        <1>       mov    [cdev], al
14712 00010AA6 C0E302              <1>       shl    bl, 2 ; *4 (dword offset)
14713 00010AA9 8B83[E08D0100]      <1>       mov    eax, [ebx+OF_SIZE]
14714 00010AAF A3[55040300]        <1>       mov    [i.size], eax ; file size
14715 00010AB4 8D83[B88D0100]      <1>       lea    eax, [ebx+OF_POINTER] ; 12/10/2016
14716 00010ABA A3[74030300]        <1>       mov    [u.fofp], eax
14717 00010ABF 8B83[688D0100]      <1>       mov    eax, [ebx+OF_FCLUSTER]
14718 00010AC5 C0EB02              <1>       shr    bl, 2 ; /4 (byte offset)
14719                              <1>       ; 17/04/2021
14720 00010AC8 F8                  <1>       clc
14721                              <1> getf4:
14722 00010AC9 C3                  <1>       retn
14723                              <1> getf5:
14724                              <1>       ; 17/04/2021
14725                              <1>       ; (following code is disasbled as temporary)
14726                              <1>       ;
14727                              <1>       ;; get device number
14728                              <1>       ;and   bl, 7Fh ; 1 to 7Fh
14729                              <1>       ;dec   bl ; 0 based (0 to 7Eh)
14730                              <1>       ;mov   al, [ebx+DEV_DRIVER]
14731                              <1>       ;mov   ch, [ebx+DEV_ACCESS]
14732                              <1>       ;mov   cl, [ebx+DEV_OPENMODE]
14733                              <1>       ;and   ch, 0FEh ; reset bit 0 ; dev_close
14734 00010ACA F9                  <1>       stc ; cf = 1
14735 00010ACB C3                  <1>       retn
14736                              <1>
14737                              <1> trans_addr_nmbp:
14738                              <1>       ; 18/10/2015
14739                              <1>       ; 12/10/2015
14740 00010ACC 8B2D[7C030300]      <1>       mov    ebp, [u.namep]
14741                              <1> trans_addr_nm:
14742                              <1>       ; Convert virtual (pathname) address to physical address
14743                              <1>       ; (Retro UNIX 386 v1 feature only !)
14744                              <1>       ; 18/10/2015
14745                              <1>       ; 12/10/2015 (u.pnbase & u.pncount has been removed from code)
14746                              <1>       ; 02/07/2015
14747                              <1>       ; 17/06/2015
14748                              <1>       ; 16/06/2015
14749                              <1>       ;
14750                              <1>       ; INPUTS:
14751                              <1>       ;      ebp = pathname address (virtual) ; [u.namep]
14752                              <1>       ;      [u.pgdir] = user's page directory
14753                              <1>       ; OUTPUT:
14754                              <1>       ;       esi = physical address of the pathname
14755                              <1>       ;      ecx = remain byte count in the page
14756                              <1>       ;
14757                              <1>       ; (Modified registers: EAX, EBX, ECX, EDX, ESI)
14758                              <1>       ;
14759 00010AD2 833D[BC030300]00    <1>         cmp     dword [u.ppgdir], 0  ; /etc/init ? (sysexec)
14760 00010AD9 7618                <1>       jna    short trans_addr_nmk ; the caller is os kernel;
14761                              <1>                                   ; it is already physical address
14762 00010ADB 50                  <1>       push   eax
14763 00010ADC 89EB                <1>       mov    ebx, ebp ; [u.namep] ; pathname address (virtual)
14764 00010ADE E87F53FFFF          <1>              call   get_physical_addr ; get physical address
14765 00010AE3 7204                <1>       jc     short tr_addr_nm_err
14766                              <1>       ; 18/10/2015
14767                              <1>       ; eax = physical address
14768                              <1>       ; cx = remain byte count in page (1-4096)
14769                              <1>              ; 12/10/2015 (cx = [u.pncount])
14770 00010AE5 89C6                <1>       mov    esi, eax ; 12/10/2015 (esi=[u.pnbase])
14771 00010AE7 58                  <1>       pop    eax
14772 00010AE8 C3                  <1>       retn
14773                              <1>
14774                              <1> tr_addr_nm_err:
14775 00010AE9 A3[C8030300]        <1>       mov    [u.error], eax
14776                              <1>       ;pop   eax
14777 00010AEE E9BDC8FFFF          <1>       jmp    error
14778                              <1>
14779                              <1> trans_addr_nmk:
14780                              <1>       ; 12/10/2015
14781                              <1>       ; 02/07/2015
14782 00010AF3 8B35[7C030300]      <1>       mov    esi, [u.namep]  ; [u.pnbase]
14783 00010AF9 66B90010            <1>       mov    cx, PAGE_SIZE ; 4096 ; [u.pncount]
14784 00010AFD C3                  <1>       retn
14785                              <1>
14786                              <1> sysbreak:
14787                              <1>       ; 18/10/2015
14788                              <1>       ; 07/10/2015
14789                              <1>       ; 23/06/2015 (Retro UNIX 386 v1 - Beginning)
14790                              <1>       ; 20/06/2013 - 24/03/2014 (Retro UNIX 8086 v1)
14791                              <1>       ;
14792                              <1>       ; 'sysbreak' sets the programs break points.
14793                              <1>       ; It checks the current break point (u.break) to see if it is
14794                              <1>       ; between "core" and the stack (sp). If it is, it is made an
14795                              <1>       ; even address (if it was odd) and the area between u.break
14796                              <1>       ; and the stack is cleared. The new breakpoint is then put
14797                              <1>       ; in u.break and control is passed to 'sysret'.
14798                              <1>       ;
14799                              <1>       ; Calling sequence:
14800                              <1>       ;      sysbreak; addr
14801                              <1>       ; Arguments: -
14802                              <1>       ;
14803                              <1>       ; Inputs: u.break - current breakpoint
```

```
14804                                  <1>       ; Outputs: u.break - new breakpoint
14805                                  <1>       ;      area between old u.break and the stack (sp) is cleared.
14806                                  <1>       ; ..........................................................
14807                                  <1>       ;
14808                                  <1>       ; Retro UNIX 8086 v1 modification:
14809                                  <1>       ;      The user/application program puts breakpoint address
14810                                  <1>       ;       in BX register as 'sysbreak' system call argument.
14811                                  <1>       ;      (argument transfer method 1)
14812                                  <1>       ;
14813                                  <1>       ;  NOTE: Beginning of core is 0 in Retro UNIX 8086 v1 !
14814                                  <1>       ;      ((!'sysbreak' is not needed in Retro UNIX 8086 v1!))
14815                                  <1>       ;  NOTE:
14816                                  <1>       ;      'sysbreak' clears extended part (beyond of previous
14817                                  <1>       ;      'u.break' address) of user's memory for original unix's
14818                                  <1>       ;      'bss' compatibility with Retro UNIX 8086 v1 (19/11/2013)
14819                                  <1>
14820                                  <1>             ; mov u.break,r1 / move users break point to r1
14821                                  <1>             ; cmp r1,$core / is it the same or lower than core?
14822                                  <1>             ; blos 1f / yes, 1f
14823                                  <1>       ; 23/06/2015
14824 00010AFE 8B2D[90030300]         <1>       mov    ebp, [u.break] ; virtual address (offset)
14825                                  <1>       ;and   ebp, ebp
14826                                  <1>       ;jz    short sysbreak_3
14827                                  <1>       ; Retro UNIX 386 v1 NOTE: u.break points to virtual address !!!
14828                                  <1>       ; (Even break point address is not needed for Retro UNIX 386 v1)
14829 00010B04 8B15[5C030300]         <1>       mov    edx, [u.sp] ; kernel stack at the beginning of sys call
14830 00010B0A 83C20C                 <1>       add    edx, 12 ; EIP -4-> CS -4-> EFLAGS -4-> ESP (user)
14831                                  <1>       ; 07/10/2015
14832 00010B0D 891D[90030300]         <1>       mov    [u.break], ebx ; virtual address !!!
14833                                  <1>       ;
14834 00010B13 3B1A                   <1>       cmp    ebx, [edx] ; compare new break point with
14835                                  <1>                         ; with top of user's stack (virtual!)
14836 00010B15 7323                   <1>       jnb    short sysbreak_3
14837                                  <1>             ; cmp r1,sp / is it the same or higher
14838                                  <1>                     ; / than the stack?
14839                                  <1>             ; bhis 1f / yes, 1f
14840 00010B17 89DE                   <1>       mov    esi, ebx
14841 00010B19 29EE                   <1>       sub    esi, ebp ; new break point - old break point
14842 00010B1B 761D                   <1>       jna    short sysbreak_3
14843                                  <1>       ;push  ebx
14844                                  <1> sysbreak_1:
14845 00010B1D 89EB                   <1>       mov    ebx, ebp
14846 00010B1F E83E53FFFF             <1>       call   get_physical_addr ; get physical address
14847 00010B24 72C3                   <1>       jc     tr_addr_nm_err
14848                                  <1>       ; 18/10/2015
14849 00010B26 89C7                   <1>       mov    edi, eax
14850 00010B28 29C0                   <1>       sub    eax, eax ; 0
14851                                  <1>              ; ECX = remain byte count in page (1-4096)
14852 00010B2A 39CE                   <1>       cmp    esi, ecx
14853 00010B2C 7302                   <1>       jnb    short sysbreak_2
14854 00010B2E 89F1                   <1>       mov    ecx, esi
14855                                  <1> sysbreak_2:
14856 00010B30 29CE                   <1>       sub    esi, ecx
14857 00010B32 01CD                   <1>       add    ebp, ecx
14858 00010B34 F3AA                   <1>       rep    stosb
14859 00010B36 09F6                   <1>       or     esi, esi
14860 00010B38 75E3                   <1>       jnz    short sysbreak_1
14861                                  <1>       ;
14862                                  <1>             ; bit $1,r1 / is it an odd address
14863                                  <1>             ; beq 2f / no, its even
14864                                  <1>             ; clrb (r1)+ / yes, make it even
14865                                  <1>       ; 2: / clear area between the break point and the stack
14866                                  <1>             ; cmp r1,sp / is it higher or same than the stack
14867                                  <1>             ; bhis 1f / yes, quit
14868                                  <1>             ; clr (r1)+ / clear word
14869                                  <1>             ; br 2b / go back
14870                                  <1>       ;pop   ebx
14871                                  <1> sysbreak_3: ; 1:
14872                                  <1>       ;mov   [u.break], ebx ; virtual address !!!
14873                                  <1>             ; jsr r0,arg; u.break / put the "address"
14874                                  <1>                   ; / in u.break (set new break point)
14875                                  <1>             ; br sysret4 / br sysret
14876 00010B3A E991C8FFFF             <1>       jmp    sysret
14877                                  <1>
14878                                  <1> sysseek: ; / moves read write pointer in an fsp entry
14879                                  <1>       ; 06/11/2016 - TRDOS 386 (TRDOS v2.0)
14880                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
14881                                  <1>       ; 07/07/2013 - 05/08/2013 (Retro UNIX 8086 v1)
14882                                  <1>       ;
14883                                  <1>       ; 'sysseek' changes the r/w pointer of (3rd word of in an
14884                                  <1>       ; fsp entry) of an open file whose file descriptor is in u.r0.
14885                                  <1>       ; The file descriptor refers to a file open for reading or
14886                                  <1>       ; writing. The read (or write) pointer is set as follows:
14887                                  <1>       ;      * if 'ptrname' is 0, the pointer is set to offset.
14888                                  <1>       ;      * if 'ptrname' is 1, the pointer is set to its
14889                                  <1>       ;        current location plus offset.
14890                                  <1>       ;      * if 'ptrname' is 2, the pointer is set to the
14891                                  <1>       ;        size of file plus offset.
14892                                  <1>       ; The error bit (e-bit) is set for an undefined descriptor.
14893                                  <1>       ;
14894                                  <1>       ; Calling sequence:
14895                                  <1>       ;      sysseek; offset; ptrname
14896                                  <1>       ; Arguments:
14897                                  <1>       ;      offset - number of bytes desired to move
14898                                  <1>       ;               the r/w pointer
14899                                  <1>       ;      ptrname - a switch indicated above
14900                                  <1>       ;
14901                                  <1>       ; Inputs: r0 - file descriptor
14902                                  <1>       ; Outputs: -
14903                                  <1>       ; ..........................................................
14904                                  <1>       ;
14905                                  <1>       ; Retro UNIX 8086 v1 modification:
14906                                  <1>       ;       'sysseek' system call has three arguments; so,
14907                                  <1>       ;      * 1st argument, file descriptor is in BX (BL) register
14908                                  <1>       ;      * 2nd argument, offset is in CX register
```

```
14909                                  <1>       ;      * 3rd argument, ptrname/switch is in DX (DL) register
14910                                  <1>
14911 00010B3F E821000000              <1>       call   seektell
14912                                  <1>       ; EAX = Current R/W pointer of the file
14913                                  <1>       ; EBX = [u.fofp]
14914                                  <1>       ; [u.base] = offset (ECX input)
14915                                  <1>
14916 00010B44 0305[84030300]          <1>       add    eax, [u.base]
14917 00010B4A 8903                    <1>       mov    [ebx], eax
14918 00010B4C E97FC8FFFF              <1>       jmp    sysret
14919                                  <1>
14920                                  <1> systell: ; / get the r/w pointer
14921                                  <1>       ; 06/11/2016 - TRDOS 386 (TRDOS v2.0) - temporary !-
14922                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
14923                                  <1>       ; 07/07/2013 - 05/08/2013 (Retro UNIX 8086 v1)
14924                                  <1>       ;
14925                                  <1>       ; Retro UNIX 8086 v1 modification:
14926                                  <1>       ; ! 'systell' does not work in original UNIX v1,
14927                                  <1>       ;          it returns with error !
14928                                  <1>       ; Inputs: r0 - file descriptor
14929                                  <1>       ; Outputs: r0 - file r/w pointer
14930                                  <1>
14931                                  <1>       ;xor   ecx, ecx ; 0
14932 00010B51 BA01000000              <1>       mov    edx, 1 ; 05/08/2013
14933                                  <1>       ;call  seektell
14934 00010B56 E810000000              <1>       call   seektell0 ; 05/08/2013
14935                                  <1>       ;; 06/11/2016
14936                                  <1>       ;; mov eax, [ebx]
14937 00010B5B A3[64030300]            <1>       mov    [u.r0], eax
14938 00010B60 E96BC8FFFF              <1>       jmp    sysret
14939                                  <1>
14940                                  <1> ; Original unix v1 'systell' system call:
14941                                  <1>             ; jsr r0,seektell
14942                                  <1>             ; br error4
14943                                  <1>
14944                                  <1> seektell:
14945                                  <1>       ; 06/11/2016 - TRDOS 386 (TRDOS v2.0)
14946                                  <1>       ; 03/01/2016
14947                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
14948                                  <1>       ; 07/07/2013 - 05/08/2013 (Retro UNIX 8086 v1)
14949                                  <1>       ;
14950                                  <1>       ; 'seektell' puts the arguments from sysseek and systell
14951                                  <1>       ; call in u.base and u.count. It then gets the i-number of
14952                                  <1>       ; the file from the file descriptor in u.r0 and by calling
14953                                  <1>       ; getf. The i-node is brought into core and then u.count
14954                                  <1>       ; is checked to see it is a 0, 1, or 2.
14955                                  <1>       ; If it is 0 - u.count stays the same
14956                                  <1>       ;          1 - u.count = offset (u.fofp)
14957                                  <1>       ;         2 - u.count = i.size (size of file)
14958                                  <1>       ;
14959                                  <1>       ; !! Retro UNIX 8086 v1 modification:
14960                                  <1>       ;      Argument 1, file descriptor is in BX;
14961                                  <1>       ;      Argument 2, offset is in CX;
14962                                  <1>       ;      Argument 3, ptrname/switch is in DX register.
14963                                  <1>       ;
14964                                  <1>       ; ((Return -> eax = base for offset (position= base+offset))
14965                                  <1>       ;
14966 00010B65 890D[84030300]          <1>       mov    [u.base], ecx ; offset
14967                                  <1> seektell0:
14968 00010B6B 8915[88030300]          <1>       mov    [u.count], edx
14969                                  <1>       ; EBX = file descriptor (file number)
14970 00010B71 E80CFFFFFF              <1>       call   getf1
14971                                  <1>       ; EAX = First cluster of the file
14972                                  <1>       ; EBX = File number (Open file number)
14973                                  <1>       ; [u.fofp] = Pointer to File pointer
14974                                  <1>       ; [i.size] = File size
14975                                  <1>
14976 00010B76 09C0                    <1>       or     eax, eax
14977 00010B78 7514                    <1>       jnz    short seektell1
14978                                  <1>
14979 00010B7A B80A000000              <1>       mov    eax, ERR_FILE_NOT_OPEN
14980 00010B7F A3[64030300]            <1>       mov    [u.r0], eax
14981 00010B84 A3[C8030300]            <1>       mov    dword [u.error], eax ; 'file not open !'
14982 00010B89 E922C8FFFF              <1>       jmp    error
14983                                  <1>
14984                                  <1> seektell1:
14985 00010B8E 8B1D[74030300]          <1>       mov    ebx, [u.fofp]
14986 00010B94 803D[88030300]01        <1>       cmp    byte [u.count], 1
14987 00010B9B 7705                    <1>       ja     short seektell2
14988 00010B9D 7409                    <1>       je     short seektell3
14989 00010B9F 31C0                    <1>       xor    eax, eax
14990 00010BA1 C3                      <1>       retn
14991                                  <1>
14992                                  <1> seektell2:
14993 00010BA2 A1[55040300]            <1>       mov    eax, [i.size]
14994 00010BA7 C3                      <1>       retn
14995                                  <1>
14996                                  <1> seektell3:
14997 00010BA8 8B03                    <1>       mov    eax, [ebx]
14998 00010BAA C3                      <1>       retn
14999                                  <1>
15000                                  <1> sysintr: ; / set interrupt handling
15001                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
15002                                  <1>       ; 07/07/2013 (Retro UNIX 8086 v1)
15003                                  <1>       ;
15004                                  <1>       ; 'sysintr' sets the interrupt handling value. It puts
15005                                  <1>       ; argument of its call in u.intr then branches into 'sysquit'
15006                                  <1>       ; routine. u.tty is checked if to see if a control tty exists.
15007                                  <1>       ; If one does the interrupt character in the tty buffer is
15008                                  <1>       ; cleared and 'sysret'is called. If one does not exits
15009                                  <1>       ; 'sysret' is just called.
15010                                  <1>       ;
15011                                  <1>       ; Calling sequence:
15012                                  <1>       ;     sysintr; arg
15013                                  <1>       ; Argument:
```

```
15014                                  <1>       ;      arg - if 0, interrupts (ASCII DELETE) are ignored.
15015                                  <1>       ;          - if 1, intterupts cause their normal result
15016                                  <1>       ;             i.e force an exit.
15017                                  <1>       ;          - if arg is a location within the program,
15018                                  <1>       ;            control is passed to that location when
15019                                  <1>       ;            an interrupt occurs.
15020                                  <1>       ; Inputs: -
15021                                  <1>       ; Outputs: -
15022                                  <1>       ; ..........................................................
15023                                  <1>       ;
15024                                  <1>       ; Retro UNIX 8086 v1 modification:
15025                                  <1>       ;       'sysintr' system call sets u.intr to value of BX
15026                                  <1>       ;      then branches into sysquit.
15027                                  <1>       ;
15028 00010BAB 66891D[AA030300]       <1>       mov    [u.intr], bx
15029                                  <1>              ; jsr r0,arg; u.intr / put the argument in u.intr
15030                                  <1>              ; br 1f / go into quit routine
15031 00010BB2 E919C8FFFF             <1>       jmp    sysret
15032                                  <1>
15033                                  <1> sysquit:
15034                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
15035                                  <1>       ; 07/07/2013 (Retro UNIX 8086 v1)
15036                                  <1>       ;
15037                                  <1>       ; 'sysquit' turns off the quit signal. it puts the argument of
15038                                  <1>       ; the call in u.quit. u.tty is checked if to see if a control
15039                                  <1>       ; tty exists. If one does the interrupt character in the tty
15040                                  <1>       ; buffer is cleared and 'sysret'is called. If one does not exits
15041                                  <1>       ; 'sysret' is just called.
15042                                  <1>       ;
15043                                  <1>       ; Calling sequence:
15044                                  <1>       ;      sysquit; arg
15045                                  <1>       ; Argument:
15046                                  <1>       ;      arg - if 0, this call diables quit signals from the
15047                                  <1>       ;            typewriter (ASCII FS)
15048                                  <1>       ;          - if 1, quits are re-enabled and cause execution to
15049                                  <1>       ;            cease and a core image to be produced.
15050                                  <1>       ;             i.e force an exit.
15051                                  <1>       ;          - if arg is an addres in the program,
15052                                  <1>       ;            a quit causes control to sent to that
15053                                  <1>       ;            location.
15054                                  <1>       ; Inputs: -
15055                                  <1>       ; Outputs: -
15056                                  <1>       ; ..........................................................
15057                                  <1>       ;
15058                                  <1>       ; Retro UNIX 8086 v1 modification:
15059                                  <1>       ;       'sysquit' system call sets u.quit to value of BX
15060                                  <1>       ;      then branches into 'sysret'.
15061                                  <1>       ;
15062 00010BB7 66891D[AC030300]       <1>       mov    [u.quit], bx
15063 00010BBE E90DC8FFFF             <1>       jmp    sysret
15064                                  <1>              ; jsr r0,arg; u.quit / put argument in u.quit
15065                                  <1>       ;1:
15066                                  <1>              ; mov u.ttyp,r1 / move pointer to control tty buffer
15067                                  <1>                              ; / to r1
15068                                  <1>              ; beq sysret4 / return to user
15069                                  <1>              ; clrb 6(r1) / clear the interrupt character
15070                                  <1>                       ; / in the tty buffer
15071                                  <1>              ; br sysret4 / return to user
15072                                  <1>
15073                                  <1> anyi:
15074                                  <1>       ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
15075                                  <1>       ; Major Modification!
15076                                  <1>       ; TRDOS 386 does not permit to delete a file while it is open
15077                                  <1>       ; The role of 'anyi' procedure has beeen changed to ensure that.
15078                                  <1>       ;
15079                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
15080                                  <1>       ; 25/04/2013 (Retro UNIX 8086 v1)
15081                                  <1>       ;
15082                                  <1>       ; 'anyi' is called if a file deleted while open.
15083                                  <1>       ; "anyi" checks to see if someone else has opened this file.
15084                                  <1>       ;
15085                                  <1>       ; INPUTS ->
15086                                  <1>       ;    r1 - contains an i-number
15087                                  <1>       ;    fsp - start of table containing open files
15088                                  <1>       ;
15089                                  <1>       ; OUTPUTS ->
15090                                  <1>       ;    "deleted" flag set in fsp entry of another occurrence of
15091                                  <1>       ;        this file and r2 points 1st word of this fsp entry.
15092                                  <1>       ;    if file not found - bit in i-node map is cleared
15093                                  <1>       ;                        (i-node is freed)
15094                                  <1>       ;                all blocks related to i-node are freed
15095                                  <1>       ;               all flags in i-node are cleared
15096                                  <1>       ; ((AX = R1)) input
15097                                  <1>       ;
15098                                  <1>       ;    (Retro UNIX Prototype : 02/12/2012, UNIXCOPY.ASM)
15099                                  <1>       ;    ((Modified registers: eDX, eCX, eBX, eSI, eDI, eBP))
15100                                  <1>       ;
15101                                  <1>       ; / r1 contains an i-number
15102                                  <1>
15103                                  <1>       ; TRDOS 386 (06/10/2016)
15104                                  <1>       ;
15105                                  <1>       ; INPUT:
15106                                  <1>       ;     EAX = First Cluster
15107                                  <1>       ;      DL = Logical DOS Drive Number
15108                                  <1>       ;
15109                                  <1>       ; OUTPUT:
15110                                  <1>       ;     CF = 1 -> EBX = File Handle/Number/Index
15111                                  <1>       ;     CF = 0 -> EBX = 0
15112                                  <1>       ;
15113                                  <1>       ; Modified Registers: EBX
15114                                  <1>
15115 00010BC3 31DB                   <1>       xor    ebx, ebx
15116                                  <1> anyi_0:
15117 00010BC5 80BB[9A8D0100]00       <1>       cmp    byte [ebx+OF_MODE], 0 ; 0 = empty entry
15118 00010BCC 770A                   <1>       ja     short anyi_2 ; 1 (r), 2 (w) or 3 (r&w)
```

```
                                <1> anyi_1:
00010BCE FEC3                   <1>         inc     bl
00010BD0 80FB0A                 <1>         cmp     bl, OPENFILES ; max. count of open files
00010BD3 72F0                   <1>         jb      short anyi_0
00010BD5 31C0                   <1>         xor     eax, eax
00010BD7 C3                     <1>         retn
                                <1> anyi_2:
00010BD8 3A93[908D0100]         <1>         cmp     dl, [ebx+OF_DRIVE]
00010BDE 75EE                   <1>         jne     short anyi_1
00010BE0 66C1E302               <1>         shl     bx, 2 ; *4 (dword offset)
00010BE4 3B83[688D0100]         <1>         cmp     eax, [ebx+OF_FCLUSTER]
00010BEA 7406                   <1>         je      short anyi_3
00010BEC 66C1EB02               <1>         shr     bx, 2 ; /4 (byte offset)
00010BF0 EBDC                   <1>         jmp     short anyi_1
                                <1> anyi_3:
00010BF2 66C1EB02               <1>         shr     bx, 2 ; /4 (bytes offset) (index)
00010BF6 F9                     <1>         stc
00010BF7 C3                     <1>         retn
                                <1>
                                <1> ; Retro UNIX 386 v1 Kernel (v0.2) - SYS9.INC
                                <1> ; Last Modification: 09/12/2015
                                <1>
                                <1> syssleep:
                                <1>         ; 29/06/2015 - (Retro UNIX 386 v1)
                                <1>         ; 11/06/2014 - (Retro UNIX 8086 v1)
                                <1>         ;
                                <1>         ; Retro UNIX 8086 v1 feature only
                                <1>         ; (INPUT -> none)
                                <1>         ;
00010BF8 0FB61D[B3030300]       <1>         movzx  ebx, byte [u.uno] ; process number
00010BFF 8AA3[7F000300]         <1>         mov    ah, [ebx+p.ttyc-1] ; current/console tty
00010C05 E881190000             <1>         call   sleep
00010C0A E9C1C7FFFF             <1>         jmp    sysret
                                <1>
                                <1> _vp_clr:
                                <1>         ; Reset/Clear Video Page
                                <1>         ;
                                <1>         ; 30/06/2015 - (Retro UNIX 386 v1)
                                <1>         ; 21/05/2013 - 30/10/2013(Retro UNIX 8086 v1) (U0.ASM)
                                <1>         ;
                                <1>         ; Retro UNIX 8086 v1 feature only !
                                <1>         ;
                                <1>         ; INPUTS ->
                                <1>         ;   BH = video page number
                                <1>         ;
                                <1>         ; OUTPUT ->
                                <1>         ;   none
                                <1>         ; ((Modified registers: eAX, BH, eCX, eDX, eSI, eDI))
                                <1>         ;
                                <1>         ; 04/12/2013
00010C0F 28C0                   <1>         sub    al, al
                                <1>         ; al = 0 (clear video page)
                                <1>         ; bh = video page ; 13/05/2016
00010C11 B407                   <1>         mov    ah, 07h
                                <1>         ; ah = 7 (attribute/color)
00010C13 6631C9                 <1>         xor    cx, cx ; 0, left upper column (cl) & row (cl)
00010C16 66BA4F18               <1>         mov    dx, 184Fh ; right lower column & row (dl=24, dh=79)
00010C1A E89513FFFF             <1>         call   _scroll_up
                                <1>         ; bh = video page
00010C1F 6631D2                 <1>         xor    dx, dx ; 0 (cursor position)
00010C22 E9D316FFFF             <1>         jmp    _set_cpos
                                <1>
                                <1> sysmsg:
                                <1>         ; 07/12/2020
                                <1>         ; 05/12/2020
                                <1>         ; 13/05/2016
                                <1>         ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>         ; 01/07/2015 - 11/11/2015 (Retro UNIX 386 v1)
                                <1>         ; Print user-application message on user's console tty
                                <1>         ;
                                <1>         ; Input -> EBX = Message address
                                <1>         ;         ECX = Message length (max. 255)
                                <1>         ;         DL = Color (IBM PC Rombios color attributes)
                                <1>         ;
00010C27 81F9FF000000           <1>         cmp    ecx, MAX_MSG_LEN ; 255
00010C2D 0F879DC7FFFF           <1>         ja     sysret ; nothing to do with big message size
00010C33 08C9                   <1>         or     cl, cl
00010C35 0F8495C7FFFF           <1>         jz     sysret
00010C3B 20D2                   <1>         and    dl, dl
00010C3D 7502                   <1>         jnz    short sysmsg0
00010C3F B207                   <1>         mov    dl, 07h ; default color
                                <1>                ; (black background, light gray character)
                                <1> sysmsg0:
00010C41 891D[84030300]         <1>         mov    [u.base], ebx
00010C47 8815[D7800100]         <1>         mov    [ccolor], dl ; color attributes
00010C4D 89E5                   <1>         mov    ebp, esp
00010C4F 31DB                   <1>         xor    ebx, ebx ; 0
00010C51 891D[8C030300]         <1>         mov    [u.nread], ebx ; 0
                                <1>         ;
00010C57 381D[C6030300]         <1>         cmp    [u.kcall], bl ; 0
00010C5D 776F                   <1>         ja     short sysmsgk ; Temporary (01/07/2015)
                                <1>         ;
00010C5F 890D[88030300]         <1>         mov    [u.count], ecx
                                <1>         ;inc   ecx ; + 00h ; ASCIIZ
                                <1>         ;
                                <1>         ; 07/12/2020
                                <1>         ;add   ecx, 3
00010C65 6683C103               <1>         add    cx, 3
00010C69 80E1FC                 <1>         and    cl, ~3  ; not 3
                                <1>         ;
00010C6C 29CC                   <1>         sub    esp, ecx
00010C6E 89E7                   <1>         mov    edi, esp
00010C70 89E6                   <1>         mov    esi, esp
00010C72 66891D[C4030300]       <1>         mov    [u.pcount], bx ; reset page (phy. addr.) counter
                                <1>         ; 11/11/2015
```

```
15224 00010C79 8A25[94030300]      <1>       mov   ah, [u.ttyp] ; recent open tty
15225                              <1>       ; 0 = none
15226 00010C7F FECC                <1>       dec   ah
15227 00010C81 790C                <1>       jns   short sysmsg1
15228 00010C83 8A1D[B3030300]      <1>       mov   bl, [u.uno] ; process number
15229 00010C89 8AA3[7F000300]      <1>       mov   ah, [ebx+p.ttyc-1] ; user's (process's) console tty
15230                              <1> sysmsg1:
15231 00010C8F 8825[96030300]      <1>       mov   [u.ttyn], ah
15232                              <1> sysmsg2:
15233 00010C95 E848080000          <1>       call  cpass
15234 00010C9A 7416                <1>       jz    short sysmsg5
15235 00010C9C AA                  <1>       stosb
15236 00010C9D 20C0                <1>       and   al, al
15237 00010C9F 75F4                <1>       jnz   short sysmsg2
15238                              <1> sysmsg3:
15239 00010CA1 80FC07              <1>       cmp   ah, 7 ; tty number
15240 00010CA4 7711                <1>       ja    short sysmsg6 ; serial port
15241 00010CA6 E83E000000          <1>       call  print_cmsg ; 05/12/2020
15242                              <1> sysmsg4:
15243 00010CAB 89EC                <1>       mov   esp, ebp
15244 00010CAD E91EC7FFFF          <1>       jmp   sysret
15245                              <1> sysmsg5:
15246 00010CB2 C60700              <1>       mov   byte [edi], 0
15247 00010CB5 EBEA                <1>       jmp   short sysmsg3
15248                              <1> sysmsg6:
15249 00010CB7 8A06                <1>       mov   al, [esi]
15250 00010CB9 E8CD180000          <1>       call  sndc
15251 00010CBE 72EB                <1>       jc    short sysmsg4
15252 00010CC0 803E00              <1>       cmp   byte [esi], 0  ; 0 is stop character
15253 00010CC3 76E6                <1>       jna   short sysmsg4
15254 00010CC5 46                  <1>       inc   esi
15255 00010CC6 8A25[96030300]      <1>       mov   ah, [u.ttyn]
15256 00010CCC EBE9                <1>       jmp   short sysmsg6
15257                              <1>
15258                              <1> sysmsgk: ; Temporary (01/07/2015)
15259                              <1>       ; The message has been sent by Kernel (ASCIIZ string)
15260                              <1>       ; (ECX -character count- will not be considered)
15261 00010CCE 8B35[84030300]      <1>       mov   esi, [u.base]
15262 00010CD4 8A25[D6800100]      <1>       mov   ah, [ptty] ; present/current screen (video page)
15263 00010CDA 8825[96030300]      <1>       mov   [u.ttyn], ah
15264 00010CE0 C605[C6030300]00    <1>       mov   byte [u.kcall], 0
15265 00010CE7 EBB8                <1>       jmp   short sysmsg3
15266                              <1>
15267                              <1> print_cmsg:
15268                              <1>       ; 08/12/2020
15269                              <1>       ; 07/12/2020
15270                              <1>       ; 05/12/2020
15271                              <1>       ; 18/11/2017
15272                              <1>       ; 13/05/2016 - TRDOS 386 (TRDOS v2.0)
15273                              <1>       ; 01/07/2015 (Retro UNIX 386 v1)
15274                              <1>       ;
15275                              <1>       ; print message (on user's console tty)
15276                              <1>       ;     with requested color
15277                              <1>       ;
15278                              <1>       ; INPUTS:
15279                              <1>       ;     esi = message address
15280                              <1>       ;     [u.ttyn] = tty number (0 to 7)
15281                              <1>       ;     [ccolor] = color attributes (IBM PC BIOS colors)
15282                              <1>       ;
15283                              <1>       ; Modified registers: eax, ebx, ecx, edx, esi, edi
15284                              <1>       ; (ebp must be preserved)
15285                              <1>
15286                              <1>       ;mov  bh, ah
15287 00010CE9 8A3D[96030300]      <1>       mov   bh, [u.ttyn]
15288 00010CEF 8A1D[D7800100]      <1>       mov   bl, [ccolor] ; * ; 05/12/2020
15289                              <1>
15290                              <1>       ; 05/12/2020
15291 00010CF5 803D[04120300]00    <1>       cmp   byte [pmi32], 0 ; is vbios's 32 bit pmi enabled ?
15292 00010CFC 772E                <1>       ja    short pcmsg5 ; yes
15293                              <1> pcmsg1:
15294                              <1>       ; 08/12/2020
15295 00010CFE 8A1D[D7800100]      <1>       mov   bl, [ccolor] ; * (video.s 'u11'&'beep' change BL)
15296                              <1>
15297 00010D04 AC                  <1>       lodsb
15298 00010D05 20C0                <1>       and   al, al  ; 0
15299 00010D07 743A                <1>       jz    short pcmsg2
15300                              <1> pcmsg7:
15301 00010D09 56                  <1>       push  esi
15302                              <1>       ;mov  bl, [ccolor] ; * (video.s 'u11'&'beep' change BL)
15303                              <1>       ; 05/12/2020
15304                              <1>       ;;mov  bh, [u.ttyn]
15305                              <1>       ;call  _write_tty
15306                              <1>       ;pop  esi
15307                              <1>       ;jmp  short pcmsg1
15308                              <1> ;pcmsg2:
15309                              <1>       ;retn
15310                              <1>
15311                              <1>       ; 07/12/2020
15312 00010D0A 803D[BA6A0000]03    <1>       cmp   byte [CRT_MODE], 3
15313 00010D11 7708                <1>       ja    short pcmsg4
15314                              <1> pcmsg3:
15315 00010D13 E85E15FFFF          <1>       call  _write_tty_m3
15316 00010D18 5E                  <1>       pop   esi
15317 00010D19 EBE3                <1>       jmp   short pcmsg1
15318                              <1> pcmsg4:
15319 00010D1B 803D[BA6A0000]07    <1>       cmp   byte [CRT_MODE], 7
15320 00010D22 76EF                <1>       jna   short pcmsg3
15321 00010D24 E85B22FFFF          <1>       call  vga_write_teletype
15322 00010D29 5E                  <1>       pop   esi
15323 00010D2A EBD2                <1>       jmp   short pcmsg1
15324                              <1> pcmsg5:
15325                              <1>       ; 07/12/2020
15326 00010D2C 803D[BA6A0000]07    <1>       cmp   byte [CRT_MODE], 7
15327 00010D33 76C9                <1>       jna   short pcmsg1
15328                              <1>
```

```
15329                                  <1>       ; 05/12/2020
15330                                  <1>       ; writing message by using
15331                                  <1>       ; VESA VBE3 video bios protected mode interface
15332                                  <1>
15333 00010D35 B40E                   <1>       mov    ah, 0Eh
15334                                  <1> pcmsg6:
15335 00010D37 AC                     <1>       lodsb
15336 00010D38 20C0                   <1>       and    al, al  ; 0
15337 00010D3A 7407                   <1>       jz     short pcmsg2
15338                                  <1>       ; bh = video page
15339                                  <1>       ; ah = 0Eh
15340                                  <1>       ; al = character
15341                                  <1>       ; bl = color
15342 00010D3C E85B0CFFFF             <1>       call   int10h_32bit_pmi
15343 00010D41 EBF4                   <1>       jmp    short pcmsg6
15344                                  <1> pcmsg2:
15345 00010D43 C3                     <1>       retn
15346                                  <1>
15347                                  <1> sysgeterr:
15348                                  <1>       ; 09/12/2015
15349                                  <1>       ; 21/09/2015 - (Retro UNIX 386 v1 feature only!)
15350                                  <1>       ; Get last error number or page fault count
15351                                  <1>       ; (for debugging)
15352                                  <1>       ;
15353                                  <1>       ; Input -> EBX = return type
15354                                  <1>       ;         0 = last error code (which is in 'u.error')
15355                                  <1>       ;         FFFFFFFFh = page fault count for running process
15356                                  <1>       ;         FFFFFFFEh = total page fault count
15357                                  <1>       ;         1 .. FFFFFFFDh = undefined
15358                                  <1>       ;
15359                                  <1>       ; Output -> EAX = last error number or page fault count
15360                                  <1>       ;         (depending on EBX input)
15361                                  <1>       ;
15362 00010D44 21DB                   <1>       and    ebx, ebx
15363 00010D46 750B                   <1>       jnz    short glerr_2
15364                                  <1> glerr_0:
15365 00010D48 A1[C8030300]           <1>       mov    eax, [u.error]
15366                                  <1> glerr_1:
15367 00010D4D A3[64030300]           <1>       mov    [u.r0], eax
15368 00010D52 C3                     <1>       retn
15369                                  <1> glerr_2:
15370 00010D53 43                     <1>       inc    ebx ; FFFFFFFFh -> 0, FFFFFFFEh -> FFFFFFFFh
15371 00010D54 74FD                   <1>       jz     short glerr_2 ; page fault count for process
15372 00010D56 43                     <1>       inc    ebx ; FFFFFFFFh -> 0
15373 00010D57 75EF                   <1>       jnz    short glerr_0
15374 00010D59 A1[68050300]           <1>       mov    eax, [PF_Count] ; total page fault count
15375 00010D5E EBED                   <1>         jmp     short glerr_1
15376                                  <1> glerr_3:
15377 00010D60 A1[CC030300]           <1>       mov    eax, [u.pfcount]
15378 00010D65 EBE6                   <1>       jmp    short glerr_1
15379                                  <1>
15380                                  <1> load_and_run_file:
15381                                  <1>       ; 18/11/2017
15382                                  <1>       ; 22/01/2017
15383                                  <1>       ; 04/01/2017, 07/01/2017
15384                                  <1>       ; 24/10/2016
15385                                  <1>       ; 24/04/2016, 02/05/2016, 03/05/2016, 06/05/2016
15386                                  <1>       ; 23/04/2016 (TRDOS 386 = TRDOS v2.0)
15387                                  <1>       ; 23/10/2015 (Retro UNIX 386 v1, 'sysexec')
15388                                  <1>       ; 23/06/2015 (Retro UNIX 386 v1 - Beginning)
15389                                  <1>       ; 03/06/2013 - 06/12/2013 (Retro UNIX 8086 v1)
15390                                  <1>       ; EAX = First Cluster number
15391                                  <1>       ; EDX = File Size
15392                                  <1>       ; ESI = Argument list address
15393                                  <1>       ; [argc] = argument count
15394                                  <1>       ; [u.nread] = argument list length
15395                                  <1>       ; [esp] = return address to the caller (*)
15396                                  <1>       ;
15397 00010D67 8935[4C040300]         <1>       mov    [argv], esi
15398 00010D6D 8915[55040300]         <1>       mov    [i.size], edx
15399 00010D73 A3[51040300]           <1>       mov    [ii], eax
15400                                  <1>
15401                                  <1>       ;sti   ; 07/01/2017
15402                                  <1>       ;mov   eax, [k_page_dir]
15403                                  <1>       ;mov   [u.pgdir], eax
15404 00010D78 31C0                   <1>       xor    eax, eax ; clc ; *** ; 04/01/2017
15405                                  <1>       ;mov   [u.r0], eax ; 0 ; 07/01/2017
15406                                  <1>
15407                                  <1>       ; 06/05/2016
15408                                  <1>       ; Set 'sysexit' return order to MainProg
15409                                  <1>       ;
15410 00010D7A 58                     <1>       pop    eax ; * 'loc_load_and_run_file_8:' address
15411                                  <1>       ;; 22/01/2017
15412                                  <1>       ;;cli ; 07/01/2017
15413 00010D7B 8B25[44800100]         <1>       mov    esp, [tss.esp0]
15414                                  <1>       ;
15415                                  <1>       ; 'loc_load_run_file_8' address has
15416                                  <1>       ; 'jmp loc_file_rw_restore_retn' instruction
15417                                  <1>       ; 'loc_file_rw_restore_retn:' will return to
15418                                  <1>       ; [mainprog_return_addr]
15419                                  <1>       ; just after 'call command_interpreter'
15420                                  <1>       ;
15421 00010D81 68[23700000]           <1>       push   _end_of_mainprog ; we must not return to here !
15422 00010D86 FF35[288D0100]         <1>       push   dword [mainprog_return_addr]
15423 00010D8C 89E5                   <1>       mov    ebp, esp ; **
15424                                  <1>       ;
15425 00010D8E 9C                     <1>       pushfd  ; EFLAGS      ; IRETD ; ***
15426 00010D8F 6A08                   <1>       push   KCODE ; cs    ; IRETD
15427 00010D91 50                     <1>       push   eax ; * (eip) ; IRETD
15428 00010D92 8925[5C030300]         <1>       mov    [u.sp], esp
15429                                  <1>       ;mov   byte [u.quant], time_count
15430 00010D98 1E                     <1>       push   ds
15431 00010D99 06                     <1>       push   es
15432 00010D9A 0FA0                   <1>       push   fs
15433 00010D9C 0FA8                   <1>       push   gs
```

```
15434                                  <1>       ;mov   eax, [u.r0]
15435 00010D9E 29C0                    <1>       sub    eax, eax
15436 00010DA0 60                      <1>       pushad
15437 00010DA1 68[D0D30000]            <1>       push   sysret
15438                                  <1>       ;push  sysrel1 ; 07/01/2017
15439 00010DA6 8925[60030300]          <1>       mov    [u.usp], esp
15440                                  <1>       ;
15441 00010DAC E845060000              <1>       call   wswap ; Save MainProg (process 1) 'u' structure
15442                                  <1>                    ; and registers for return (from program)
15443 00010DB1 89EC                    <1>       mov    esp, ebp ; **
15444                                  <1>       ;;22/01/2017
15445                                  <1>       ;;sti ; 07/01/2017
15446 00010DB3 50                      <1>       push   eax  ; * 'loc_load_and_run_file_8:' address
15447                                  <1>       ;
15448                                  <1>       ;;; 02/05/2016
15449                                  <1>       ;;; Create a new process (parent: MainProg)
15450 00010DB4 31F6                    <1>       xor    esi, esi
15451                                  <1> cnpm_1: ; search p.stat table for unused process number
15452 00010DB6 46                      <1>       inc    esi
15453 00010DB7 80BE[AF000300]00        <1>       cmp    byte [esi+p.stat-1], 0 ; SFREE
15454                                  <1>                           ; is process active, unused, dead
15455 00010DBE 760B                    <1>       jna    short cnpm_2 ; it's unused so branch
15456 00010DC0 6683FE10                <1>       cmp    si, nproc    ; all processes checked
15457 00010DC4 72F0                    <1>       jb     short cnpm_1    ; no, branch back
15458 00010DC6 E9E062FFFF              <1>       jmp    panic
15459                                  <1> cnpm_2:
15460 00010DCB A1[B8030300]            <1>       mov    eax, [u.pgdir] ; page directory of MainProg
15461 00010DD0 A3[BC030300]            <1>       mov    [u.ppgdir], eax ; parent's page directory
15462 00010DD5 E89B4CFFFF              <1>       call   allocate_page
15463 00010DDA 0F82CB62FFFF            <1>       jc     panic
15464                                  <1>       ; EAX = UPAGE (user structure page) address
15465 00010DE0 A3[B4030300]            <1>       mov    [u.upage], eax ; memory page for 'user' struct (child)
15466 00010DE5 89F7                    <1>       mov    edi, esi
15467 00010DE7 66C1E702                <1>       shl    di, 2
15468 00010DEB 8987[BC000300]          <1>       mov    [edi+p.upage-4], eax ; memory page for 'user' struct
15469 00010DF1 E8F04CFFFF              <1>       call   clear_page ; 03/05/2016
15470                                  <1>       ;movzx eax, byte [p.ttyc] ; console tty (for MainProg)
15471 00010DF6 6629C0                  <1>       sub    ax, ax ; 0
15472 00010DF9 668986[7F000300]        <1>       mov     [esi+p.ttyc-1], ax ; al - set child's console tty
15473                                  <1>                                ; ah - reset child's wait channel
15474 00010E00 89F0                    <1>       mov    eax, esi
15475 00010E02 A2[B3030300]            <1>       mov    [u.uno], al ; child process number
15476 00010E07 FE86[AF000300]          <1>         inc     byte [esi+p.stat-1] ; 1, SRUN
15477 00010E0D 66D1E6                  <1>       shl    si, 1 ; multiply si by 2 to get index into p.pid table
15478 00010E10 66FF05[4E030300]        <1>       inc    word [mpid] ; increment m.pid; get a new process name
15479 00010E17 66A1[4E030300]          <1>       mov    ax, [mpid]
15480 00010E1D 668986[1E000300]        <1>       mov    [esi+p.pid-2], ax ; put new process name
15481                                  <1>                              ; in child process' name slot
15482                                  <1>       ;mov   ax, [p.pid]  ; get process name of MainProg
15483 00010E24 66B80100                <1>       mov    ax, 1
15484 00010E28 668986[3E000300]        <1>       mov    [esi+p.ppid-2], ax ; put parent process name
15485                                  <1>                                ; in parent process slot for child
15486 00010E2F 6648                    <1>       dec    ax ; 0
15487 00010E31 66A3[94030300]          <1>       mov    [u.ttyp], ax ; 0
15488                                  <1>       ;;;
15489 00010E37 A1[51040300]            <1>       mov    eax,  [ii]
15490                                  <1>       ; Retro UNIX 386 v1, 'sysexec' (u2.s)
15491 00010E3C E84A170000              <1>       call   iopen
15492                                  <1>       ; 06/06/2016
15493 00010E41 C605[A9030300]01        <1>       mov    byte [u.pri], 1 ; normal priority
15494                                  <1>       ;
15495 00010E48 EB16                    <1>       jmp    short sysexec_7 ; 02/05/2016
15496                                  <1>
15497                                  <1> sysexec_6:
15498                                  <1>       ; 19/11/2017
15499                                  <1>       ; 18/11/2017
15500                                  <1>       ; 14/11/2017
15501                                  <1>       ; 13/11/2017
15502 00010E4A 8925[4C040300]          <1>       mov    [argv], esp ; *!* ; start address of argument list
15503                                  <1>
15504                                  <1>       ; 04/01/2017
15505                                  <1>       ; 24/10/2016
15506                                  <1>       ;;02/05/2016
15507                                  <1>       ; 23/04/2016 (TRDOS 386)
15508                                  <1>       ; 18/10/2015 ('sysexec_6')
15509                                  <1>       ; 23/06/2015
15510 00010E50 A1[B8030300]            <1>       mov    eax, [u.pgdir] ; physical address of page directory
15511                                  <1>       ;cmp   eax, [k_page_dir] ; TRDOS MainProg ?
15512                                  <1>       ;je    short sysexec_7
15513                                  <1>       ; 19/11/2017
15514 00010E55 8B1D[BC030300]          <1>       mov    ebx, [u.ppgdir] ; phy addr of the parent's page dir
15515 00010E5B E8454DFFFF              <1>       call   deallocate_page_dir
15516                                  <1> sysexec_7:
15517 00010E60 E8754CFFFF              <1>       call   make_page_dir
15518 00010E65 0F824062FFFF            <1>       jc     panic  ; allocation error
15519                                  <1>                     ; after a deallocation would be nonsence !?
15520                                  <1>       ; 24/07/2015
15521                                  <1>       ; map kernel pages (1st 4MB) to PDE 0
15522                                  <1>       ;     of the user's page directory
15523                                  <1>       ;     (It is needed for interrupts!)
15524                                  <1>       ; 18/10/2015
15525 00010E6B 8B15[A8800100]          <1>       mov    edx, [k_page_dir] ; Kernel's page directory
15526 00010E71 8B02                    <1>       mov    eax, [edx] ; physical address of
15527                                  <1>                         ; kernel's first page table (1st 4 MB)
15528                                  <1>                         ; (PDE 0 of kernel's page directory)
15529 00010E73 8B15[B8030300]          <1>       mov    edx, [u.pgdir]
15530 00010E79 8902                    <1>       mov    [edx], eax ; PDE 0 (1st 4MB)
15531                                  <1>       ;
15532                                  <1>       ; 20/07/2015
15533 00010E7B BB00004000              <1>       mov    ebx, CORE ; start address = 0 (virtual) + CORE
15534                                  <1>       ; 18/10/2015
15535 00010E80 BE[3C040300]            <1>       mov    esi, pcore ; physical start address
15536                                  <1> sysexec_8:
15537 00010E85 B907000000              <1>       mov    ecx, PDE_A_USER + PDE_A_WRITE + PDE_A_PRESENT
15538 00010E8A E8694CFFFF              <1>       call   make_page_table
```

```
15539 00010E8F 0F821662FFFF        <1>       jc    panic
15540                              <1>       ;mov  ecx, PTE_A_USER + PTE_A_WRITE + PTE_A_PRESENT
15541 00010E95 E86C4CFFFF          <1>       call  make_page ; make new page, clear and set the pte
15542 00010E9A 0F820B62FFFF        <1>       jc    panic
15543                              <1>       ;
15544 00010EA0 8906                <1>       mov   [esi], eax ; 24/06/2015
15545                              <1>       ; ebx = virtual address (24/07/2015)
15546 00010EA2 E8BB4FFFFF          <1>       call  add_to_swap_queue
15547                              <1>       ; 18/10/2015
15548 00010EA7 81FE[40040300]      <1>       cmp   esi, ecore ; user's stack (last) page ?
15549 00010EAD 740C                <1>       je    short sysexec_9 ; yes
15550 00010EAF BE[40040300]        <1>       mov   esi, ecore  ; physical address of the last page
15551                              <1>       ; 20/07/2015
15552 00010EB4 BB00F0FFFF          <1>       mov   ebx, (ECORE - PAGE_SIZE) + CORE
15553                              <1>       ; ebx = virtual end address + segment base address - 4K
15554 00010EB9 EBCA                <1>       jmp   short sysexec_8
15555                              <1> sysexec_9:
15556                              <1>       ; 19/11/2017
15557                              <1>       ; 24/04/2016 (TRDOS 386 = TRDOS v2.0)
15558                              <1>       ; 25/06/2015, 26/08/2015, 18/10/2015
15559                              <1>       ; move arguments from kernel stack to [ecore]
15560                              <1>       ; (argument list/line will be copied from kernel stack
15561                              <1>       ; frame to the last (stack) page of user's core memory)
15562                              <1>       ; 18/10/2015
15563 00010EBB 8B3D[40040300]      <1>       mov   edi, [ecore]
15564 00010EC1 81C700100000        <1>       add   edi, PAGE_SIZE
15565                              <1>       ; 19/11/2017
15566 00010EC7 83EF04              <1>       sub   edi, 4
15567 00010ECA C70700000000        <1>       mov   dword [edi], 0
15568 00010ED0 89FB                <1>       mov   ebx, edi
15569                              <1>       ;
15570 00010ED2 0FB705[4A040300]    <1>       movzx eax, word [argc]
15571 00010ED9 09C0                <1>       or    eax, eax
15572 00010EDB 7445                <1>       jz    short sysexec_13 ; 19/11/2017
15573                              <1>       ;jnz  short sysexec_10
15574                              <1>       ;mov  ebx, edi
15575                              <1>       ;sub  ebx, 4
15576                              <1>       ;mov  [ebx], eax ; 0
15577                              <1>       ;jmp  short sysexec_13
15578                              <1> sysexec_10:
15579 00010EDD 8B0D[8C030300]      <1>       mov   ecx, [u.nread]
15580                              <1>       ; 13/11/2017
15581                              <1>       ;mov  esi, TextBuffer ; 'load_and_execute_file'
15582                              <1>       ;mov  esi, esp     ; 'sysexec'
15583 00010EE3 8B35[4C040300]      <1>       mov   esi, [argv] ; 24/04/2016 (TRDOS 386  = TRDOS v2.0)
15584                              <1>       ;sub  edi, ecx ; page end address - argument list length
15585 00010EE9 29CB                <1>       sub   ebx, ecx ; 19/11/2017
15586 00010EEB 89C2                <1>       mov   edx, eax
15587 00010EED FEC2                <1>       inc   dl ; argument count + 1 for argc value
15588 00010EEF C0E202              <1>       shl   dl, 2  ; 4 * (argument count + 1)
15589                              <1>       ;mov  ebx, edi
15590 00010EF2 89DF                <1>       mov   edi, ebx ; 19//11/2017
15591 00010EF4 80E3FC              <1>       and   bl, 0FCh ; 32 bit (dword) alignment
15592 00010EF7 29D3                <1>       sub   ebx, edx
15593 00010EF9 89FA                <1>       mov   edx, edi
15594 00010EFB F3A4                <1>       rep   movsb
15595 00010EFD 89D6                <1>       mov   esi, edx
15596 00010EFF 89DF                <1>       mov   edi, ebx
15597 00010F01 BA00F0BFFF          <1>       mov   edx, ECORE - PAGE_SIZE ; virtual addr. of the last page
15598 00010F06 2B15[40040300]      <1>       sub   edx, [ecore] ; difference (virtual - physical)
15599 00010F0C AB                  <1>       stosd ; eax = argument count
15600                              <1> sysexec_11:
15601 00010F0D 89F0                <1>       mov   eax, esi
15602 00010F0F 01D0                <1>       add   eax, edx
15603 00010F11 AB                  <1>       stosd  ; eax = virtual address
15604                              <1>       ;dec  byte [argc]
15605 00010F12 66FF0D[4A040300]    <1>       dec   word [argc] ; 14/11/2017
15606 00010F19 7407                <1>       jz    short sysexec_13
15607                              <1> sysexec_12:
15608 00010F1B AC                  <1>       lodsb
15609 00010F1C 20C0                <1>       and   al, al
15610 00010F1E 75FB                <1>       jnz   short sysexec_12
15611 00010F20 EBEB                <1>       jmp   short sysexec_11
15612                              <1> sysexec_13:
15613                              <1>       ; 24/10/2016
15614                              <1>       ; 24/04/2016 - TRDOS 386 (TRDOS v2.0)
15615                              <1>       ; 23/06/2015 - 19/10/2015 (Retro UNIX 386 v1, 'sysexec_13')
15616                              <1>       ;
15617                              <1>       ; moving arguments to [ecore] is OK here..
15618                              <1>       ;
15619                              <1>       ; ebx = beginning addres of argument list pointers
15620                              <1>             ;      in user's stack
15621 00010F22 2B1D[40040300]      <1>       sub   ebx, [ecore]
15622 00010F28 81C300F0BFFF        <1>       add     ebx, (ECORE - PAGE_SIZE)
15623                              <1>                ; end of core - 4096 (last page)
15624                              <1>                ; (virtual address)
15625 00010F2E 891D[4C040300]      <1>       mov   [argv], ebx
15626 00010F34 891D[90030300]      <1>       mov   [u.break], ebx ; available user memory
15627                              <1>       ;
15628 00010F3A 29C0                <1>       sub   eax, eax
15629 00010F3C C705[88030300]2000- <1>       mov   dword [u.count], 32 ; Executable file header size
15629 00010F44 0000                <1>
15630 00010F46 C705[74030300]-     <1>       mov   dword [u.fofp], u.off
15630 00010F4C [80030300]          <1>
15631 00010F50 A3[80030300]        <1>       mov   [u.off], eax ; 0
15632 00010F55 A3[84030300]        <1>       mov   [u.base], eax ; 0, start of user's core (virtual)
15633                              <1>       ; 24/10/2016
15634 00010F5A A0[6E810100]        <1>       mov   al, [Current_Drv]
15635 00010F5F A2[46030300]        <1>       mov   [cdev], al
15636                              <1>       ;
15637 00010F64 A1[51040300]        <1>       mov   eax, [ii] ; Fist Cluster of the Program (PRG) file
15638                              <1>       ; EAX = First cluster of the executable file
15639 00010F69 E80A010000          <1>       call  readi
15640                              <1>
15641 00010F6E 8B0D[90030300]      <1>       mov   ecx, [u.break] ; top of user's stack (physical addr.)
```

```
15642 00010F74 890D[88030300]      <1>       mov    [u.count], ecx ; save for overrun check
15643                              <1>       ;
15644 00010F7A 8B0D[8C030300]      <1>       mov    ecx, [u.nread]
15645 00010F80 890D[90030300]      <1>       mov    [u.break], ecx ; virtual address (offset from start)
15646 00010F86 80F920              <1>       cmp    cl, 32
15647 00010F89 7540                <1>       jne    short sysexec_15
15648                              <1>       ;:
15649                              <1>       ; Retro UNIX 386 v1 (32 bit) executable file header format
15650 00010F8B 8B35[3C040300]      <1>       mov    esi, [pcore] ; start address of user's core memory
15651                              <1>                           ; (phys. start addr. of the exec. file)
15652 00010F91 AD                  <1>       lodsd
15653 00010F92 663DEB1E            <1>       cmp    ax, 1EEBh ; EBH, 1Eh -> jump to +32
15654 00010F96 7533                <1>       jne    short sysexec_15
15655 00010F98 AD                  <1>       lodsd
15656 00010F99 89C1                <1>       mov    ecx, eax ; text (code) section size
15657 00010F9B AD                  <1>       lodsd
15658 00010F9C 01C1                <1>       add    ecx, eax ; + data section size (initialized data)
15659 00010F9E 89CB                <1>       mov    ebx, ecx
15660 00010FA0 AD                  <1>       lodsd
15661 00010FA1 01C3                <1>       add    ebx, eax ; + bss section size (for overrun checking)
15662 00010FA3 3B1D[88030300]      <1>       cmp    ebx, [u.count]
15663 00010FA9 7711                <1>       ja     short sysexec_14  ; program overruns stack !
15664                              <1>       ;
15665                              <1>       ; add bss section size to [u.break]
15666 00010FAB 0105[90030300]      <1>       add    [u.break], eax
15667                              <1>       ;
15668 00010FB1 83E920              <1>       sub    ecx, 32  ; header size (already loaded)
15669                              <1>       ;cmp   ecx, [u.count]
15670                              <1>       ;jnb   short sysexec_16
15671 00010FB4 890D[88030300]      <1>       mov    [u.count], ecx ; required read count
15672 00010FBA EB29                <1>       jmp    short sysexec_16
15673                              <1> sysexec_14:
15674                              <1>       ; insufficient (out of) memory
15675 00010FBC C705[C8030300]0400- <1>       mov    dword [u.error], ERR_MINOR_IM ; 1
15675 00010FC4 0000                <1>
15676 00010FC6 E9E5C3FFFF          <1>       jmp    error
15677                              <1> sysexec_15:
15678 00010FCB 8B15[55040300]      <1>         mov  edx, [i.size] ; file size
15679 00010FD1 29CA                <1>       sub    edx, ecx ; file size - loaded bytes
15680 00010FD3 7626                <1>       jna    short sysexec_17 ; no need to next read
15681 00010FD5 01D1                <1>       add    ecx, edx ; [i.size]
15682 00010FD7 3B0D[88030300]      <1>       cmp    ecx, [u.count] ; overrun check (!)
15683 00010FDD 77DD                <1>       ja     short sysexec_14
15684 00010FDF 8915[88030300]      <1>       mov    [u.count], edx
15685                              <1> sysexec_16:
15686 00010FE5 A1[51040300]        <1>       mov    eax, [ii] ; first cluster
15687 00010FEA E889000000          <1>       call   readi
15688 00010FEF 8B0D[8C030300]      <1>       mov    ecx, [u.nread]
15689 00010FF5 010D[90030300]      <1>       add    [u.break], ecx
15690                              <1> sysexec_17:
15691 00010FFB A1[51040300]        <1>       mov    eax, [ii] ; first cluster
15692 00011000 E886150000          <1>       call   iclose
15693 00011005 31C0                <1>       xor     eax, eax
15694 00011007 FEC0                <1>       inc    al
15695 00011009 66A3[AA030300]      <1>       mov    [u.intr], ax ; 1 (interrupt/time-out is enabled)
15696 0001100F 66A3[AC030300]      <1>       mov    [u.quit], ax ; 1 ('crtl+brk' signal is enabled)
15697 00011015 833D[BC030300]00    <1>         cmp  dword [u.ppgdir], 0  ; is the caller MainProg (kernel) ?
15698 0001101C 770C                <1>       ja     short sysexec_18 ; no, the caller is user process
15699                              <1>       ; If the caller is kernel (MainProg), 'sysexec' will come here
15700 0001101E 8B15[A8800100]      <1>       mov    edx, [k_page_dir] ; kernel's page directory
15701 00011024 8915[BC030300]      <1>       mov    [u.ppgdir], edx ; next time 'sysexec' must not come here
15702                              <1> sysexec_18:
15703                              <1>       ; 02/05/2016
15704                              <1>       ; 24/04/2016 (TRDOS 386 = TRDOS v2.0)
15705                              <1>       ; 18/10/2015 (Retro UNIX 386 v1)
15706                              <1>       ; 05/08/2015
15707                              <1>       ; 29/07/2015
15708                              <1>
15709                              <1> ;     ; **** arguments list test start - 19/11/2017
15710                              <1> ;     mov    ebp, [argv]
15711                              <1> ;     sub    ebp, ECORE - 4096
15712                              <1> ;     add    ebp, [ecore]
15713                              <1> ;
15714                              <1> ;     mov    ebx, [ebp]
15715                              <1> ;     mov    [argc], bx
15716                              <1> ;     add    ebp, 4
15717                              <1> ;     mov    byte [ccolor], 1Fh
15718                              <1> ;_zx0:
15719                              <1> ;     cmp    word [argc], 0
15720                              <1> ;     jna    short _zx2
15721                              <1> ;_zx1:
15722                              <1> ;     push   ebp
15723                              <1> ;     mov    esi, [ebp]
15724                              <1> ;
15725                              <1> ;     sub    esi, ECORE - 4096
15726                              <1> ;     add    esi, [ecore]
15727                              <1> ;
15728                              <1> ;     call   print_cmsg
15729                              <1> ;
15730                              <1> ;     dec    word [argc]
15731                              <1> ;     jz     short _zx2
15732                              <1> ;
15733                              <1> ;     mov    al, '.'
15734                              <1> ;     mov    bl, 07h
15735                              <1> ;     mov    bh, [u.ttyn]
15736                              <1> ;     call   _write_tty
15737                              <1> ;
15738                              <1> ;     pop    ebp
15739                              <1> ;     add    ebp, 4
15740                              <1> ;     jmp    short _zx1
15741                              <1> ;_zx2:
15742                              <1> ;     pop    ebp
15743                              <1> ;     mov    byte [ccolor], 07h
15744                              <1> ;     mov    eax, 1
15745                              <1> ;     ; **** arguments list test stop
```

```
15746                                  <1> ;     Test result is OK! (there is not a wrong thing) - 19/11/2017
15747                                  <1>
15748 0001102A 8B2D[4C040300]          <1>       mov   ebp, [argv] ; user's stack pointer must point to argument
15749                                  <1>                         ; list pointers (argument count)
15750 00011030 FA                      <1>       cli
15751 00011031 8B25[44800100]          <1>       mov   esp, [tss.esp0] ; ring 0 (kernel) stack pointer
15752                                  <1>       ;mov  esp, [u.sp] ; Restore Kernel stack
15753                                  <1>                         ; for this process
15754                                  <1>       ;add  esp, 20 ; --> EIP, CS, EFLAGS, ESP, SS
15755                                  <1>       ;xor  eax, eax ; 0
15756 00011037 FEC8                    <1>       dec   al ; eax = 0
15757                                  <1>       ;mov  edx, UDATA
15758                                  <1>       ; 18/11/2017
15759 00011039 6A23                    <1>       push  UDATA ; user's stack segment
15760                                  <1>       ;push edx
15761 0001103B 55                      <1>       push  ebp ; user's stack pointer
15762                                  <1>                 ; (points to number of arguments)
15763                                  <1>
15764                                  <1>       ; 04/01/2017
15765                                  <1>       ; MainProg comes here while [sysflg]= 0FFh
15766                                  <1>       ; (but sysexec comes here while [sysflg]= 0)
15767 0001103C C605[5B030300]00        <1>       mov   byte [sysflg], 0 ; 04/01/2017
15768                                  <1>                              ; (timer_int sysflg control)
15769 00011043 FB                      <1>       sti
15770 00011044 9C                      <1>       pushfd ; EFLAGS
15771                                  <1>              ; Set IF for enabling interrupts in user mode
15772                                  <1>       ;or   dword [esp], 200h
15773                                  <1>       ;
15774                                  <1>       ;mov  bx, UCODE
15775                                  <1>       ;push bx ; user's code segment
15776 00011045 6A1B                    <1>       push  UCODE
15777                                  <1>       ;push 0
15778 00011047 50                      <1>       push  eax ; EIP (=0) - start address -
15779 00011048 8925[5C030300]          <1>       mov   [u.sp], esp ; 29/07/2015
15780                                  <1>       ; 05/08/2015
15781                                  <1>       ; Remedy of a General Protection Fault during 'iretd' is here !
15782                                  <1>       ; ('push dx' would cause to general protection fault,
15783                                  <1>       ; after 'pop ds' etc.)
15784                                  <1>       ;
15785                                  <1>       ;; push dx ; ds (UDATA)
15786                                  <1>       ;; push dx ; es (UDATA)
15787                                  <1>       ;; push dx ; fs (UDATA)
15788                                  <1>       ;; push dx ; gs (UDATA)
15789                                  <1>       ;
15790                                  <1>       ; This is a trick to prevent general protection fault
15791                                  <1>       ; during 'iretd' intruction at the end of 'sysrele' (in u1.s):
15792 0001104E 66BA2300                <1>       mov   dx, UDATA ; 19/11/2017
15793 00011052 8EC2                    <1>       mov   es, dx ; UDATA
15794 00011054 06                      <1>       push  es ; ds (UDATA)
15795 00011055 06                      <1>       push  es ; es (UDATA)
15796 00011056 06                      <1>       push  es ; fs (UDATA)
15797 00011057 06                      <1>       push  es ; gs (UDATA)
15798 00011058 66BA1000                <1>       mov   dx, KDATA
15799 0001105C 8EC2                    <1>       mov   es, dx
15800                                  <1>       ;
15801                                  <1>       ;; pushad simulation
15802 0001105E 89E5                    <1>       mov   ebp, esp ; esp before pushad
15803 00011060 50                      <1>       push  eax ; eax (0)
15804 00011061 50                      <1>       push  eax ; ecx (0)
15805 00011062 50                      <1>       push  eax ; edx (0)
15806 00011063 50                      <1>       push  eax ; ebx (0)
15807 00011064 55                      <1>       push  ebp ; esp before pushad
15808 00011065 50                      <1>       push  eax ; ebp (0)
15809 00011066 50                      <1>       push  eax ; esi (0)
15810 00011067 50                      <1>       push  eax ; edi (0)
15811                                  <1>       ;
15812 00011068 A3[64030300]            <1>       mov   [u.r0], eax ; eax = 0
15813 0001106D 8925[60030300]          <1>       mov   [u.usp], esp
15814                                  <1>
15815                                  <1>       ; 14/11/2017
15816 00011073 E95AC3FFFF              <1>       jmp   sysret0
15817                                  <1>
15818                                  <1> ;     ; 02/05/2016
15819                                  <1> ;     ;inc  byte [sysflg] ; 0FFh -> 0
15820                                  <1> ;     ;mov  byte [sysflg], 0 ; 04/01/2017
15821                                  <1> ;     movzx ebx, byte [u.uno]
15822                                  <1> ;     shl   bl, 1 ; 13/11/2017
15823                                  <1> ;     cmp   word [ebx+p.ppid-2], 1 ; MainProg
15824                                  <1> ;     ja    sysret0 ; 03/05/2016
15825                                  <1> ;     push  sysret ; *
15826                                  <1> ;     mov   [u.usp], esp
15827                                  <1> ;     call  wswap ; save child process 'u' structure and
15828                                  <1> ;                 ; registers
15829                                  <1> ;     add   dword [u.usp], 4 ; 03/05/2016
15830                                  <1> ;sysexec_19: ; 02/05/2016
15831                                  <1> ;     retn ; * 'sysret' ; byte [sysflg] -> 0FFh
15832                                  <1>
15833                                  <1> readi:
15834                                  <1>       ; 01/05/2016
15835                                  <1>       ; 25/04/2016 - TRDOS 386 (TRDOS v2.0)
15836                                  <1>       ; 20/05/2015 - Retro UNIX 386 v1
15837                                  <1>       ; 11/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
15838                                  <1>       ;
15839                                  <1>       ; Reads from a file whose the first cluster number in EAX
15840                                  <1>       ;
15841                                  <1>       ; INPUTS ->
15842                                  <1>       ;    EAX - First cluster number of the file
15843                                  <1>       ;    u.count - byte count user desires
15844                                  <1>       ;    u.base - points to user buffer
15845                                  <1>       ;    u.fofp - points to dword with current file offset
15846                                  <1>       ;    i.size - file size
15847                                  <1>       ;    cdev - logical dos drive number of the file
15848                                  <1>       ; OUTPUTS ->
15849                                  <1>       ;    u.count - cleared
15850                                  <1>       ;    u.nread - accumulates total bytes passed back
```

```
15851                                  <1>       ;
15852                                  <1>       ; ((EAX)) input/output
15853                                  <1>       ; (Retro UNIX Prototype : 14/12/2012 - 01/03/2013, UNIXCOPY.ASM)
15854                                  <1>         ; ((Modified registers: edx, ebx, ecx, esi, edi))
15855                                  <1>
15856 00011078 31D2                   <1>       xor    edx, edx ; 0
15857 0001107A 8915[8C030300]         <1>       mov    [u.nread], edx ; 0
15858 00011080 668915[C4030300]       <1>       mov    [u.pcount], dx ; 19/05/2015
15859 00011087 3915[88030300]         <1>       cmp    [u.count], edx ; 0
15860 0001108D 7701                   <1>       ja     short readi_1
15861 0001108F C3                     <1>       retn
15862                                  <1> readi_1:
15863                                  <1> dskr:
15864                                  <1>       ; 01/05/2016
15865                                  <1>       ; 25/04/2016 - TRDOS 386 (TRDOS v2.0)
15866                                  <1>       ; 24/05/2015 - 12/10/2015 (Retro UNIX 386 v1)
15867                                  <1>       ; 26/04/2013 - 03/08/2013 (Retro UNIX 8086 v1)
15868                                  <1> dskr_0:
15869 00011090 8B15[55040300]         <1>       mov    edx, [i.size]
15870 00011096 8B1D[74030300]         <1>       mov    ebx, [u.fofp]
15871 0001109C 2B13                   <1>       sub    edx, [ebx]
15872 0001109E 7647                   <1>       jna    short dskr_4
15873                                  <1>       ;
15874 000110A0 50                     <1>       push   eax ; 01/05/2016
15875 000110A1 3B15[88030300]         <1>       cmp    edx, [u.count]
15876 000110A7 7306                   <1>       jnb    short dskr_1
15877 000110A9 8915[88030300]         <1>       mov    [u.count], edx
15878                                  <1> dskr_1:
15879                                  <1>       ; EAX = First Cluster
15880                                  <1>       ; [Current_Drv] = Physical drive number
15881 000110AF E83B000000             <1>       call   mget_r
15882                                  <1>       ; NOTE: in 'mget_r', relevant sector will be read in buffer
15883                                  <1>       ; if it is not already in buffer !
15884 000110B4 BB[74050300]           <1>       mov    ebx, readi_buffer
15885 000110B9 803D[C6030300]00       <1>       cmp    byte [u.kcall], 0 ; the caller is 'namei' sign (=1)
15886 000110C0 770F                   <1>       ja     short dskr_3   ; zf=0 -> the caller is 'namei'
15887 000110C2 66833D[C4030300]00     <1>       cmp    word [u.pcount], 0
15888 000110CA 7705                   <1>       ja     short dskr_3
15889                                  <1> dskr_2:
15890                                  <1>       ; [u.base] = virtual address to transfer (as destination address)
15891 000110CC E894010000             <1>       call   trans_addr_w ; translate virtual address to physical (w)
15892                                  <1> dskr_3:
15893                                  <1>       ; EBX (r5) = system (I/O) buffer address -physical-
15894 000110D1 E8F7010000             <1>       call   sioreg
15895 000110D6 87F7                   <1>       xchg   esi, edi
15896                                  <1>       ; EDI = file (user data) offset
15897                                  <1>       ; ESI = sector (I/O) buffer offset
15898                                  <1>       ; ECX = byte count
15899 000110D8 F3A4                   <1>       rep    movsb
15900                                  <1>       ; eax = remain bytes in buffer
15901                                  <1>       ;       (check if remain bytes in the buffer > [u.pcount])
15902 000110DA 09C0                   <1>       or     eax, eax
15903 000110DC 75EE                   <1>       jnz    short dskr_2 ; (page end before system buffer end!)
15904 000110DE 58                     <1>       pop    eax  ; (first cluster number)
15905 000110DF 390D[88030300]         <1>       cmp    [u.count], ecx ; 0
15906 000110E5 77A9                   <1>       ja     short dskr_0
15907                                  <1> dskr_4:
15908 000110E7 C605[C6030300]00       <1>       mov    byte [u.kcall], 0
15909 000110EE C3                     <1>       retn
15910                                  <1>
15911                                  <1> mget_r:
15912                                  <1>       ; 24/10/2016
15913                                  <1>       ; 22/10/2016
15914                                  <1>       ; 12/10/2016
15915                                  <1>       ; 29/04/2016
15916                                  <1>       ; 25/04/2016 - TRDOS 386 (TRDOS v2.0)
15917                                  <1>       ; 03/06/2015 (Retro UNIX 386 v1, 'mget', u.5s)
15918                                  <1>       ; 22/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
15919                                  <1>       ;
15920                                  <1>       ; Get existing or (allocate) a new disk block for file
15921                                  <1>       ;
15922                                  <1>       ; INPUTS ->
15923                                  <1>       ;    [u.fofp] = file offset pointer
15924                                  <1>       ;    EAX = First Cluster
15925                                  <1>       ;    [cdev] = Logical dos drive number
15926                                  <1>       ;    ([u.off] = file offset)
15927                                  <1>       ; OUTPUTS ->
15928                                  <1>       ;    EAX = logical sector number
15929                                  <1>       ;    ESI = Logical Dos Drive Description Table address
15930                                  <1>       ;
15931                                  <1>       ; Modified registers: EDX, EBX, ECX, ESI, EDI
15932                                  <1>
15933 000110EF 8B35[74030300]         <1>       mov    esi, [u.fofp]
15934 000110F5 8B1E                   <1>       mov    ebx, [esi] ; (u.off)
15935                                  <1>
15936 000110F7 29C9                   <1>       sub    ecx, ecx
15937 000110F9 8A2D[46030300]         <1>       mov    ch, [cdev]
15938                                  <1>
15939 000110FF BE00010900             <1>       mov    esi, Logical_DOSDisks
15940 00011104 01CE                   <1>       add    esi, ecx
15941                                  <1>
15942 00011106 380D[DC8C0100]         <1>       cmp    [readi.valid], cl ; 0
15943 0001110C 7649                   <1>       jna    short mget_r_0
15944                                  <1>
15945 0001110E 3A2D[DD8C0100]         <1>       cmp    ch, [readi.drv]
15946 00011114 7541                   <1>       jne    short mget_r_0
15947                                  <1>
15948 00011116 3B05[F08C0100]         <1>       cmp    eax, [readi.fclust]
15949 0001111C 7565                   <1>       jne    short mget_r_3
15950                                  <1>
15951 0001111E 89D8                   <1>       mov    eax, ebx ; file offset
15952 00011120 668B0D[E48C0100]       <1>       mov    cx, [readi.bpc]
15953 00011127 41                     <1>       inc    ecx ; <= 65536
15954 00011128 29D2                   <1>       sub    edx, edx
15955 0001112A F7F1                   <1>       div    ecx
```

```
15956                                  <1>
15957 0001112C 8B3D[EC8C0100]          <1>       mov   edi, [readi.c_index] ; cluster index
15958                                  <1>
15959 00011132 39F8                    <1>       cmp   eax, edi
15960 00011134 757A                    <1>       jne   short mget_r_4  ; (*)
15961                                  <1>
15962                                  <1>       ; edx = byte offset in cluster (<= 65535)
15963 00011136 668915[E68C0100]        <1>       mov   [readi.offset], dx
15964 0001113D 66C1EA09                <1>       shr   dx, 9 ; / 512
15965 00011141 8815[DF8C0100]          <1>       mov   [readi.s_index], dl ; sector index in cluster (0 to spc -1)
15966                                  <1>
15967 00011147 A1[E88C0100]            <1>       mov   eax, [readi.cluster]  ; > 0 if [readi.valid] = 1
15968 0001114C 8B15[F48C0100]          <1>       mov   edx, [readi.fs_index]
15969 00011152 E99A000000              <1>       jmp   mget_r_7
15970                                  <1>
15971                                  <1> mget_r_0:
15972 00011157 882D[DD8C0100]          <1>       mov   [readi.drv], ch ; physical drive number
15973 0001115D 807E0300                <1>       cmp   byte [esi+LD_FATType], 0
15974 00011161 7707                    <1>       ja    short mget_r_1
15975 00011163 8A4E12                  <1>       mov   cl, [esi+LD_FS_BytesPerSec+1]
15976 00011166 D0E9                    <1>       shr   cl, 1 ;  ; 1 for 512 bytes, 4 for 2048 bytes
15977 00011168 EB03                    <1>       jmp   short mget_r_2
15978                                  <1> mget_r_1:
15979 0001116A 8A4E13                  <1>       mov   cl, [esi+LD_BPB+BPB_SecPerClust]
15980                                  <1> mget_r_2:
15981 0001116D 880D[DE8C0100]          <1>       mov   [readi.spc], cl  ; sectors per cluster
15982                                  <1>       ; NOTE: readi bytes per sector value is always 512 !
15983 00011173 66C1E109                <1>       shl   cx, 9 ; * 512
15984 00011177 6649                    <1>       dec   cx ; bytes per cluster - 1
15985 00011179 66890D[E48C0100]        <1>       mov   [readi.bpc], cx
15986 00011180 6629C9                  <1>       sub   cx, cx
15987                                  <1> mget_r_3:
15988 00011183 A3[F08C0100]            <1>       mov   [readi.fclust], eax ; first cluster (or FDT address)
15989 00011188 880D[DC8C0100]          <1>       mov   [readi.valid], cl ; 0
15990                                  <1>       ;mov  [readi.s_index], cl ; 0
15991                                  <1>       ;mov  [readi.offset], cx ; 0
15992 0001118E 890D[EC8C0100]          <1>       mov   [readi.c_index], ecx ; 0
15993 00011194 890D[E88C0100]          <1>       mov   [readi.cluster], ecx ; 0
15994 0001119A 890D[E08C0100]          <1>       mov   [readi.sector], ecx ; 0
15995                                  <1>
15996 000111A0 89D8                    <1>       mov   eax, ebx ; file offset
15997 000111A2 668B0D[E48C0100]        <1>       mov   cx, [readi.bpc]
15998 000111A9 41                      <1>       inc   ecx ; <= 65536
15999 000111AA 29D2                    <1>       sub   edx, edx
16000 000111AC F7F1                    <1>       div   ecx
16001                                  <1>       ;mov  edi, [readi.c_index] ; previous cluster index
16002 000111AE 29FF                    <1>       sub   edi, edi
16003                                  <1> mget_r_4:
16004 000111B0 A3[EC8C0100]            <1>       mov   [readi.c_index], eax ; cluster index
16005                                  <1>       ; edx = byte offset in cluster (<= 65535)
16006 000111B5 668915[E68C0100]        <1>       mov   [readi.offset], dx
16007 000111BC 66C1EA09                <1>       shr   dx, 9 ; / 512
16008 000111C0 8815[DF8C0100]          <1>       mov   [readi.s_index], dl ; sector index in cluster (0 to spc -1)
16009                                  <1>
16010 000111C6 89C1                    <1>       mov   ecx, eax ; current cluster index
16011 000111C8 A1[F08C0100]            <1>       mov   eax, [readi.fclust]
16012 000111CD 09C9                    <1>       or    ecx, ecx ; cluster index
16013 000111CF 741B                    <1>       jz    short mget_r_6
16014                                  <1>
16015 000111D1 39CF                    <1>       cmp   edi, ecx
16016 000111D3 7710                    <1>       ja    short mget_r_5 ; old cluster index is higher
16017 000111D5 8B15[E88C0100]          <1>       mov   edx, [readi.cluster]
16018 000111DB 21D2                    <1>       and   edx, edx
16019 000111DD 7406                    <1>       jz    short mget_r_5
16020                                  <1>       ; valid 'readi' parameters (*)
16021 000111DF 89D0                    <1>       mov   eax, edx
16022 000111E1 29F9                    <1>       sub   ecx, edi
16023 000111E3 740C                    <1>       jz    short mget_r_7
16024                                  <1> mget_r_5:
16025                                  <1>       ; EAX = Beginning cluster
16026                                  <1>       ; EDX = Sector index in disk/file section
16027                                  <1>       ;     (Only for SINGLIX file system!)
16028                                  <1>       ; ECX = Cluster sequence number after the beginning cluster
16029                                  <1>       ; ESI = Logical DOS Drive Description Table address
16030 000111E5 E85FC0FFFF              <1>       call  get_cluster_by_index
16031 000111EA 724E                    <1>       jc    short mget_r_err
16032                                  <1>       ; EAX = Cluster number
16033                                  <1> mget_r_6:
16034 000111EC A3[E88C0100]            <1>       mov   [readi.cluster], eax ; FDT number for Singlix File System
16035                                  <1> mget_r_7:
16036 000111F1 807E0300                <1>       cmp   byte [esi+LD_FATType], 0
16037 000111F5 765F                    <1>       jna   short mget_r_12
16038                                  <1>
16039 000111F7 83E802                  <1>       sub   eax, 2
16040 000111FA 0FB615[DE8C0100]        <1>       movzx edx, byte [readi.spc]
16041 00011201 F7E2                    <1>       mul   edx
16042                                  <1>
16043 00011203 034668                  <1>       add   eax, [esi+LD_DATABegin]
16044 00011206 8A15[DF8C0100]          <1>       mov   dl, [readi.s_index]
16045 0001120C 01D0                    <1>       add   eax, edx
16046                                  <1> mget_r_8:
16047                                  <1>       ; eax = logical sector number
16048 0001120E 803D[DC8C0100]00        <1>       cmp   byte [readi.valid], 0
16049 00011215 7608                    <1>       jna   short mget_r_9
16050 00011217 3B05[E08C0100]          <1>       cmp   eax, [readi.sector]
16051 0001121D 7436                    <1>       je    short mget_r_11 ; sector is already in 'readi' buffer
16052                                  <1> mget_r_9:
16053 0001121F A3[E08C0100]            <1>       mov   [readi.sector], eax
16054 00011224 BB[74050300]            <1>       mov   ebx, readi_buffer ; buffer address
16055 00011229 B901000000              <1>       mov   ecx, 1
16056                                  <1>       ; 29/04/2016
16057                                  <1>       ;xor  dl, dl
16058                                  <1>
16059                                  <1>       ; EAX = Logical sector number
16060                                  <1>       ; ECX = Sector count
```

```
16061                                  <1>       ; EBX = Buffer address
16062                                  <1>       ; (EDX = 0)
16063                                  <1>       ; ESI = Logical DOS drive description table address
16064                                  <1>
16065 0001122E E868130000              <1>       call   disk_read
16066 00011233 7314                    <1>       jnc    short mget_r_10
16067                                  <1>
16068                                  <1>       ; 22/10/2016 (15h -> 17)
16069 00011235 B811000000              <1>       mov    eax, 17 ; Drive not ready or read error !
16070                                  <1> mget_r_err:
16071 0001123A A3[C8030300]            <1>       mov    [u.error], eax
16072                                  <1>       ; 12/10/2016
16073 0001123F A3[64030300]            <1>       mov    [u.r0], eax
16074 00011244 E967C1FFFF              <1>       jmp    error
16075                                  <1> mget_r_10:
16076 00011249 C605[DC8C0100]01        <1>       mov    byte [readi.valid], 1 ; 24/10/2016
16077 00011250 A1[E08C0100]            <1>       mov    eax, [readi.sector]
16078                                  <1> mget_r_11:
16079 00011255 C3                      <1>       retn
16080                                  <1> mget_r_12:
16081                                  <1>       ; EAX = FDT number
16082                                  <1>       ; EDX = Sector index from FDT sector (0,1,2,3,4...)
16083 00011256 40                      <1>       inc    eax ; the first data sector in FS disk section
16084 00011257 8915[F48C0100]          <1>       mov    [readi.fs_index], edx
16085 0001125D 01D0                    <1>       add    eax, edx
16086 0001125F EBAD                    <1>       jmp    short mget_r_8
16087                                  <1>
16088                                  <1> trans_addr_r:
16089                                  <1>       ; 12/10/2016
16090                                  <1>       ; 02/05/2016 - TRDOS 386 (TRDOS v2.0)
16091                                  <1>       ; Translate virtual address to physical address
16092                                  <1>       ; for reading from user's memory space
16093                                  <1>       ; 04/06/2015 - 18/10/2015 (Retro UNIX 386 v1)
16094                                  <1>
16095 00011261 31D2                    <1>       xor    edx, edx ; 0 (read access sign)
16096 00011263 EB04                    <1>       jmp    short trans_addr_rw
16097                                  <1>
16098                                  <1> trans_addr_w:
16099                                  <1>       ; 12/10/2016
16100                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
16101                                  <1>       ; Translate virtual address to physical address
16102                                  <1>       ; for writing to user's memory space
16103                                  <1>       ; 04/06/2015 - 18/10/2015 (Retro UNIX 386 v1)
16104                                  <1>
16105 00011265 29D2                    <1>       sub    edx, edx
16106 00011267 FEC2                    <1>       inc    dl ; 1 (write access sign)
16107                                  <1> trans_addr_rw:
16108 00011269 50                      <1>       push   eax
16109 0001126A 53                      <1>       push   ebx
16110 0001126B 52                      <1>       push   edx ; r/w sign (in DL)
16111                                  <1>       ;
16112 0001126C 8B1D[84030300]          <1>       mov    ebx, [u.base]
16113 00011272 E8EB4BFFFF              <1>       call   get_physical_addr ; get physical address
16114 00011277 730F                    <1>       jnc    short passc_0
16115 00011279 A3[C8030300]            <1>       mov    [u.error], eax
16116 0001127E A3[64030300]            <1>       mov    [u.r0], eax ; 12/10/2016
16117                                  <1>       ;pop   edx
16118                                  <1>       ;pop   ebx
16119                                  <1>       ;pop   eax
16120 00011283 E928C1FFFF              <1>       jmp    error
16121                                  <1> passc_0:
16122 00011288 F6C202                  <1>       test   dl, PTE_A_WRITE ; writable page
16123 0001128B 5A                      <1>       pop    edx
16124 0001128C 751C                    <1>       jnz    short passc_1
16125                                  <1>
16126 0001128E 20D2                    <1>       and    dl, dl
16127 00011290 7418                    <1>       jz     short passc_1
16128                                  <1>       ; read only (duplicated) page -must be copied to a new page-
16129                                  <1>       ; EBX = linear address
16130 00011292 51                      <1>       push   ecx
16131 00011293 E8394BFFFF              <1>       call   copy_page
16132 00011298 59                      <1>       pop    ecx
16133 00011299 721E                    <1>       jc     short passc_2
16134 0001129B 50                      <1>       push   eax ; physical address of the new/allocated page
16135 0001129C E8C14BFFFF              <1>       call   add_to_swap_queue
16136 000112A1 58                      <1>       pop    eax
16137 000112A2 81E3FF0F0000            <1>       and    ebx, PAGE_OFF ; 0FFFh
16138                                  <1>       ;mov   ecx, PAGE_SIZE
16139                                  <1>       ;sub   ecx, ebx
16140 000112A8 01D8                    <1>       add    eax, ebx
16141                                  <1> passc_1:
16142 000112AA A3[C0030300]            <1>       mov    [u.pbase], eax ; physical address
16143 000112AF 66890D[C4030300]        <1>       mov    [u.pcount], cx ; remain byte count in page (1-4096)
16144 000112B6 5B                      <1>       pop    ebx
16145 000112B7 58                      <1>       pop    eax
16146 000112B8 C3                      <1>       retn
16147                                  <1> passc_2:
16148 000112B9 B804000000              <1>       mov    eax, ERR_MINOR_IM ; "Insufficient memory !" error
16149 000112BE A3[64030300]            <1>       mov    [u.r0], eax ; 12/10/2016
16150 000112C3 A3[C8030300]            <1>       mov    dword [u.error], eax
16151                                  <1>       ;pop   ebx
16152                                  <1>       ;pop   eax
16153 000112C8 E9E3C0FFFF              <1>       jmp    error
16154                                  <1>
16155                                  <1> sioreg:
16156                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
16157                                  <1>       ; 19/05/2015 - 25/07/2015 (Retro UNIX 386 v1)
16158                                  <1>       ; 12/03/2013 - 22/07/2013 (Retro UNIX 8086 v1)
16159                                  <1>       ; INPUTS ->
16160                                  <1>       ;     EBX = system buffer (data) address (r5)
16161                                  <1>       ;     [u.fofp] = pointer to file offset pointer
16162                                  <1>       ;     [u.base] = virtual address of the user buffer
16163                                  <1>       ;     [u.pbase] = physical address of the user buffer
16164                                  <1>       ;     [u.count] = byte count
16165                                  <1>       ;     [u.pcount] = byte count within page frame
```

```
16166                                  <1>       ; OUTPUTS ->
16167                                  <1>       ;     ESI = user data offset (r1)
16168                                  <1>       ;     EDI = system (I/O) buffer offset (r2)
16169                                  <1>       ;     ECX = byte count (r3)
16170                                  <1>       ;     EAX = remain bytes after byte count within page frame
16171                                  <1>       ;      (If EAX > 0, transfer will continue from the next page)
16172                                  <1>         ;
16173                                  <1>       ; ((Modified registers:  EDX))
16174                                  <1>
16175 000112CD 8B35[74030300]          <1>       mov    esi, [u.fofp]
16176 000112D3 8B3E                    <1>       mov    edi, [esi]
16177 000112D5 89F9                    <1>       mov    ecx, edi
16178 000112D7 81C900FEFFFF            <1>       or     ecx, 0FFFFFE00h
16179 000112DD 81E7FF010000            <1>       and    edi, 1FFh
16180 000112E3 01DF                    <1>       add    edi, ebx ; EBX = system buffer (data) address
16181 000112E5 F7D9                    <1>       neg    ecx
16182 000112E7 3B0D[88030300]          <1>       cmp    ecx, [u.count]
16183 000112ED 7606                    <1>       jna    short sioreg_0
16184 000112EF 8B0D[88030300]          <1>       mov    ecx, [u.count]
16185                                  <1> sioreg_0:
16186 000112F5 803D[C6030300]00        <1>       cmp    byte [u.kcall], 0
16187 000112FC 7613                    <1>       jna    short sioreg_1
16188                                  <1>        ; the caller is 'mkdir' or 'namei'
16189 000112FE A1[84030300]            <1>       mov    eax, [u.base]
16190 00011303 A3[C0030300]            <1>       mov    [u.pbase], eax ; physical address = virtual address
16191 00011308 66890D[C4030300]        <1>       mov    word [u.pcount], cx ; remain bytes in buffer (1 sector)
16192 0001130F EB0B                    <1>       jmp    short sioreg_2
16193                                  <1> sioreg_1:
16194 00011311 0FB715[C4030300]        <1>       movzx  edx, word [u.pcount]
16195 00011318 39D1                    <1>       cmp    ecx, edx
16196 0001131A 772A                    <1>       ja     short sioreg_4 ; transfer count > [u.pcount]
16197                                  <1> sioreg_2: ; 2:
16198 0001131C 31C0                    <1>       xor    eax, eax
16199                                  <1> sioreg_3:
16200 0001131E 010D[8C030300]          <1>       add    [u.nread], ecx
16201 00011324 290D[88030300]          <1>       sub    [u.count], ecx
16202 0001132A 010D[84030300]          <1>       add    [u.base], ecx
16203 00011330 010E                    <1>       add    [esi], ecx
16204 00011332 8B35[C0030300]          <1>       mov    esi, [u.pbase]
16205 00011338 66290D[C4030300]        <1>       sub    [u.pcount], cx
16206 0001133F 010D[C0030300]          <1>       add    [u.pbase], ecx
16207 00011345 C3                      <1>       retn
16208                                  <1> sioreg_4:
16209                                  <1>       ; transfer count > [u.pcount]
16210                                  <1>       ; (ecx > edx)
16211 00011346 89C8                    <1>       mov    eax, ecx
16212 00011348 29D0                    <1>       sub    eax, edx ; remain bytes for 1 sector (block) transfer
16213 0001134A 89D1                    <1>       mov    ecx, edx ; current transfer count = [u.pcount]
16214 0001134C EBD0                    <1>       jmp    short sioreg_3
16215                                  <1>
16216                                  <1> tswitch: ; Retro UNIX 386 v1
16217                                  <1> tswap:
16218                                  <1>       ; 16/01/2017
16219                                  <1>       ; 21/05/2016 - TRDOS 386 (TRDOS v2.0)
16220                                  <1>       ; 10/05/2015 - 01/09/2015 (Retro UNIX 386 v1)
16221                                  <1>       ; 14/04/2013 - 14/02/2014 (Retro UNIX 8086 v1)
16222                                  <1>       ; time out swap, called when a user times out.
16223                                  <1>       ; the user is put on the low priority queue.
16224                                  <1>       ; This is done by making a link from the last user
16225                                  <1>       ; on the low priority queue to him via a call to 'putlu'.
16226                                  <1>       ; then he is swapped out.
16227                                  <1>
16228                                  <1>       ; TRDOS 386 (TRDOS v2.0) modification ->  ** 21/05/2016 **
16229                                  <1>       ;     * when a high priority (event) process will be stopped
16230                                  <1>       ;      (swapped out, swithched out/off), 'tswap/tswitch' will
16231                                  <1>       ;      not add it to a run queue.
16232                                  <1>       ;      /// What for: Process may be already in a run queue,
16233                                  <1>       ;      it is unspeficied state because process might be started
16234                                  <1>       ;      by a timer event which does not regard previous priority
16235                                  <1>       ;      level and run queue of the process (for fast executing!).
16236                                  <1>       ;      After the 'run for event', process will be sequenced
16237                                  <1>       ;      to run by it's actual run queue. ///
16238                                  <1>       ;
16239                                  <1>       ; Retro UNIX 386 v1 modification ->
16240                                  <1>       ;       swap (software task switch) is performed by changing
16241                                  <1>       ;      user's page directory (u.pgdir) instead of segment change
16242                                  <1>       ;      as in Retro UNIX 8086 v1.
16243                                  <1>       ;
16244                                  <1>       ; RETRO UNIX 8086 v1 modification ->
16245                                  <1>       ;       'swap to disk' is replaced with 'change running segment'
16246                                  <1>       ;      according to 8086 cpu (x86 real mode) architecture.
16247                                  <1>       ;      pdp-11 was using 64KB uniform memory while IBM PC
16248                                  <1>       ;      compatibles was using 1MB segmented memory
16249                                  <1>       ;      in 8086/8088 times.
16250                                  <1>       ;
16251                                  <1>       ; INPUTS ->
16252                                  <1>       ;    u.uno - users process number
16253                                  <1>       ;    runq+4 - lowest priority queue
16254                                  <1>       ; OUTPUTS ->
16255                                  <1>       ;    r0 - users process number
16256                                  <1>       ;    r2 - lowest priority queue address
16257                                  <1>       ;
16258                                  <1>       ; ((AX = R0, BX = R2)) output
16259                                  <1>       ; ((Modified registers: EDX, EBX, ECX, ESI, EDI))
16260                                  <1>       ;
16261                                  <1>
16262                                  <1>       NOTE:
16263                                  <1>       ;* [u.pri] priority level is specified by run queue which is process
16264                                  <1>       ;  comes to run from.
16265                                  <1>       ;* Initial [u.pri] is 1 ('normal/regular') for programs
16266                                  <1>       ;  (which are launched by MainProg or 'sysexec'), it is changed
16267                                  <1>       ;  to 2 ('high') by timer event, if program uses 'systimer' system call.
16268                                  <1>       ;* Program (Process) also can change it's running priority
16269                                  <1>       ;  from 1 to 0 or up to 2 by using 'syspri' system call; but,
16270                                  <1>       ;  if program selects priority level 2 (high) for running, next time
```

```
16271                                  <1>       ;  it is reduced to 1 (normal/regular) because 'syspri' adds this
16272                                  <1>       ;  program to 'run for normal' queue while running duration is a bit
16273                                  <1>       ;  protected from swap/switch out immediate, behalf of other high
16274                                  <1>       ;  priority process in sequence. Program (with high priority) will not
16275                                  <1>       ;  be swapped/switched out (by timer event) before it's time quantum
16276                                  <1>       ;  will be elapsed, but, this will be temporary if program is not using
16277                                  <1>       ;  timer event function.
16278                                  <1>
16279                                  <1>       ;For example:
16280                                  <1>       ;If a process frequently gets a timer event, it runs at high priority
16281                                  <1>       ;level but when it returns from running it returns to actual run queue,
16282                                  <1>       ;not to 'run for event' queue again.
16283                                  <1>       ;'tswap' will not change the sequence at return/stop(swap out) stage.
16284                                  <1>       ;But if priority level not high (=2, 'run for event'), 'tswap/tswitch'
16285                                  <1>       ;will add the stopping process to relevant run queue according to
16286                                  <1>       ;[u.pri] priority level.
16287                                  <1>
16288                                  <1>       ; 16/01/2017
16289 0001134E BB[54030300]            <1>       mov    ebx, runq+2  ; 'runq_normal' ; normal/regular priority
16290                                  <1>       ; 21/05/2016
16291                                  <1>       ;cmp   byte [u.pri], 2     ; high priority (run for event) ?
16292                                  <1>       ;jnb   short swap
16293                                  <1>       ; 16/01/2017
16294                                  <1>       ; (Normal and also high/event priority processes will be added to
16295                                  <1>       ; normal priority run queue for ensuring circular running sequence!)
16296                                  <1>       ; (Timer interrupt or 'syspri' system call may change priority and run
16297                                  <1>       ; queue to high/event level.)
16298 00011353 803D[A9030300]00       <1>       cmp    byte [u.pri], 0
16299 0001135A 7702                  <1>       ja     short tswap_1; normal priority run queue
16300                                  <1>       ;
16301 0001135C 43                    <1>       inc    ebx
16302 0001135D 43                    <1>       inc    ebx          ; runq+4, 'runq_background', low priority
16303                                  <1> tswap_1:
16304 0001135E A0[B3030300]          <1>       mov    al, [u.uno]
16305                                  <1>                     ; movb u.uno,r1 / move users process number to r1
16306                                  <1>       ; mov  $runq+4,r2
16307                                  <1>                     ; / move lowest priority queue address to r2
16308                                  <1>       ; ebx = run queue
16309 00011363 E8FE000000            <1>       call   putlu
16310                                  <1>       ; jsr r0,putlu / create link from last user on Q to
16311                                  <1>                      ; / u.uno's user
16312                                  <1>
16313                                  <1> switch: ; Retro UNIX 386 v1
16314                                  <1> swap:
16315                                  <1>       ; 02/01/2017
16316                                  <1>       ; 21/05/2016
16317                                  <1>       ; 20/05/2016
16318                                  <1>       ; 02/05/2016
16319                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
16320                                  <1>       ; 10/05/2015 - 02/09/2015 (Retro UNIX 386 v1)
16321                                  <1>       ; 14/04/2013 - 08/03/2014 (Retro UNIX 8086 v1)
16322                                  <1>       ;
16323                                  <1>       ; 'swap' is routine that controls the swapping of processes
16324                                  <1>       ; in and out of core.
16325                                  <1>       ;
16326                                  <1>       ; TRDOS 386 (TRDOS v2.0) modification ->  ** 20/05/2016 **
16327                                  <1>       ;     * 3 different priority level is applied
16328                                  <1>       ;      (just as original unix v1)
16329                                  <1>       ;      1) high priority (event) run queue, 'runq_event'
16330                                  <1>       ;      2) normal priority (regular) run queue, 'runq_normal'
16331                                  <1>       ;      3) low priority (background) run queue, 'runq_backgroud'
16332                                  <1>       ;      'swap' code will run a process which has max. priority
16333                                  <1>       ;        (for earliest event at first)
16334                                  <1>       ;
16335                                  <1>       ; Retro UNIX 386 v1 modification ->
16336                                  <1>       ;       swap (software task switch) is performed by changing
16337                                  <1>       ;      user's page directory (u.pgdir) instead of segment change
16338                                  <1>       ;      as in Retro UNIX 8086 v1.
16339                                  <1>       ;
16340                                  <1>       ; RETRO UNIX 8086 v1 modification ->
16341                                  <1>       ;       'swap to disk' is replaced with 'change running segment'
16342                                  <1>       ;      according to 8086 cpu (x86 real mode) architecture.
16343                                  <1>       ;      pdp-11 was using 64KB uniform memory while IBM PC
16344                                  <1>       ;      compatibles was using 1MB segmented memory
16345                                  <1>       ;      in 8086/8088 times.
16346                                  <1>       ;
16347                                  <1>       ; INPUTS ->
16348                                  <1>       ;    runq table - contains processes to run.
16349                                  <1>       ;    p.link - contains next process in line to be run.
16350                                  <1>       ;    u.uno - process number of process in core
16351                                  <1>       ;    s.stack - swap stack used as an internal stack for swapping.
16352                                  <1>       ; OUTPUTS ->
16353                                  <1>       ;    (original unix v1 -> present process to its disk block)
16354                                  <1>       ;    (original unix v1 -> new process into core ->
16355                                  <1>       ;         Retro Unix 8086 v1 -> segment registers changed
16356                                  <1>       ;         for new process)
16357                                  <1>       ;    u.quant = 3 (Time quantum for a process)
16358                                  <1>       ;      ((INT 1Ch count down speed -> 18.2 times per second)
16359                                  <1>       ;    RETRO UNIX 8086 v1 will use INT 1Ch (18.2 times per second)
16360                                  <1>       ;       for now, it will swap the process if there is not
16361                                  <1>       ;       a keyboard event (keystroke) (Int 15h, function 4Fh)
16362                                  <1>       ;       or will count down from 3 to 0 even if there is a
16363                                  <1>       ;         keyboard event locking due to repetitive key strokes.
16364                                  <1>       ;       u.quant will be reset to 3 for RETRO UNIX 8086 v1.
16365                                  <1>       ;
16366                                  <1>       ; ((Modified registers: EAX, EDX, EBX, ECX, ESI, EDI))
16367                                  <1>
16368                                  <1>       ;NOTE:
16369                                  <1>       ;High priority queue is the first for selecting a process to run.
16370                                  <1>       ;If there is not a process in high priority level run queue,
16371                                  <1>       ;a process in normal priority run queue will be selected
16372                                  <1>       ;or a proces in low priority run queue will be selected if normal
16373                                  <1>       ;priority level run queue is empty.
16374                                  <1>
16375                                  <1>       ; 21/05/2016 -(3 priority levels, 3 run queues)
```

```
16376 00011368 BE[52030300]        <1>       mov    esi, runq ; 'runq_event' ; high priority, 'run for event'
16377 0001136D C605[388D0100]03    <1>       mov    byte [priority], 3 ; high priority + 1
16378 00011374 31DB                <1>       xor    ebx, ebx ; 02/01/2017
16379                              <1> swap_0: ; 1: / search runq table for highest priority process
16380 00011376 66AD                <1>       lodsw  ; mov ax, [esi], add esi+2
16381                              <1>       ;xor   ebx, ebx ; 02/05/2016
16382 00011378 6621C0              <1>       and    ax, ax ; are there any processes to run in this Q entry
16383 0001137B 750E                <1>       jnz    short swap_2
16384                              <1>       ; 21/05/2026
16385                              <1>       ; runq_normal = runq+2, runq_background = runq+4
16386 0001137D FE0D[388D0100]      <1>       dec    byte [priority] ; 3 -> 3, 2 -> 1, 1-> 0
16387 00011383 75F1                <1>       jnz    short swap_0
16388                              <1>       ;cmp   esi, runq+6  ; if zero compare address to end of table
16389                              <1>       ;jb    short swap_0 ; if not at end, go back
16390                              <1> swap_1:
16391                              <1>       ; 02/05/2016
16392                              <1>       ; 29/04/2016 (TRDOS 386 = TRDOS v2.0)
16393                              <1>       ; No user process to run...
16394                              <1>       ; Run the kernel process... MainProg: Internal Command Interpreter
16395 00011385 FEC0                <1>       inc    al ; mov al, 1  ; process number of MainProg
16396 00011387 FEC3                <1>       inc    bl ; mov bl, al ; 1
16397 00011389 EB1E                <1>       jmp    short swap_4
16398                              <1> swap_2:
16399                              <1>       ; 21/05/2016
16400 0001138B FE0D[388D0100]      <1>       dec    byte [priority] ; priority level of present user/process
16401                              <1>                              ; 0, 1, 2
16402 00011391 4E                  <1>       dec    esi
16403 00011392 4E                  <1>       dec    esi
16404                              <1>       ;
16405 00011393 88C3                <1>       mov    bl, al
16406 00011395 38E0                <1>       cmp    al, ah ; is there only 1 process in the queue to be run
16407 00011397 740A                <1>       je     short swap_3 ; yes
16408 00011399 8AA3[9F000300]      <1>       mov    ah, [ebx+p.link-1]
16409 0001139F 8826                <1>       mov    [esi], ah ; move next process in line into run queue
16410 000113A1 EB06                <1>       jmp    short swap_4
16411                              <1> swap_3:
16412 000113A3 6631D2              <1>       xor    dx, dx
16413 000113A6 668916              <1>       mov    [esi], dx ; zero the entry; no processes on the Q
16414                              <1> swap_4:
16415 000113A9 8A25[B3030300]      <1>       mov    ah, [u.uno]
16416 000113AF 38C4                <1>       cmp    ah, al ; is this process the same as the process in core?
16417 000113B1 743B                <1>       je     short swap_8 ; yes, don't have to swap
16418 000113B3 08E4                <1>       or     ah, ah  ; is the process # = 0
16419 000113B5 740D                <1>       jz     short swap_6 ; 'sysexit'
16420                              <1>       ;cmp   ah, al ;is this process the same as the process in core?
16421                              <1>       ;je    short swap_8 ; yes, don't have to swap
16422 000113B7 8925[60030300]      <1>       mov    [u.usp], esp ; return  address for 'syswait' & 'sleep'
16423 000113BD E834000000          <1>       call   wswap   ; write out core to disk
16424 000113C2 EB1C                <1>       jmp    short swap_7
16425                              <1> swap_6:
16426                              <1>       ; Deallocate memory pages belong to the process
16427                              <1>       ; which is being terminated.
16428                              <1>       ; (Retro UNIX 386 v1 modification !)
16429                              <1>       ;
16430 000113C4 53                  <1>       push   ebx
16431 000113C5 A1[B8030300]        <1>       mov    eax, [u.pgdir]  ; page directory of the process
16432 000113CA 8B1D[BC030300]      <1>       mov    ebx, [u.ppgdir] ; page directory of the parent process
16433 000113D0 E8D047FFFF          <1>       call   deallocate_page_dir
16434 000113D5 A1[B4030300]        <1>       mov    eax, [u.upage] ; 'user' structure page of the process
16435 000113DA E85E48FFFF          <1>       call   deallocate_page
16436 000113DF 5B                  <1>       pop    ebx
16437                              <1> swap_7:
16438 000113E0 C0E302              <1>       shl    bl, 2 ; * 4
16439 000113E3 8B83[BC000300]      <1>       mov    eax, [ebx+p.upage-4] ; the 'u' page of the new process
16440 000113E9 E840000000          <1>       call   rswap ; read new process into core
16441                              <1> swap_8:
16442                              <1>       ; Retro UNIX  8086 v1 modification !
16443 000113EE C605[A8030300]04    <1>       mov    byte [u.quant], time_count
16444 000113F5 C3                  <1>       retn
16445                              <1>
16446                              <1> wswap:  ; < swap out, swap to disk >
16447                              <1>       ; 28/02/2017 (fnsave)
16448                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
16449                              <1>       ; 09/05/2015 (Retro UNIX 386 v1)
16450                              <1>       ; 26/05/2013 - 08/03/2014 (Retro UNIX 8086 v1)
16451                              <1>       ; 'wswap' writes out the process that is in core onto its
16452                              <1>       ; appropriate disk area.
16453                              <1>       ;
16454                              <1>       ; Retro UNIX 386 v1 modification ->
16455                              <1>       ;       User (u) structure content and the user's register content
16456                              <1>       ;     will be copied to the process's/user's UPAGE (a page for
16457                              <1>       ;     saving 'u' structure and user registers for task switching).
16458                              <1>       ;     u.usp - points to kernel stack address which contains
16459                              <1>       ;             user's registers while entering system call.
16460                              <1>       ;     u.sp  - points to kernel stack address
16461                              <1>       ;             to return from system call -for IRET-.
16462                              <1>       ;     [u.usp]+32+16 = [u.sp]
16463                              <1>       ;     [u.usp] -> edi, esi, ebp, esp (= [u.usp]+32), ebx,
16464                              <1>       ;             edx, ecx, eax, gs, fs, es, ds, -> [u.sp].
16465                              <1>       ;
16466                              <1>       ; Retro UNIX 8086 v1 modification ->
16467                              <1>       ;       'swap to disk' is replaced with 'change running segment'
16468                              <1>       ;     according to 8086 cpu (x86 real mode) architecture.
16469                              <1>       ;     pdp-11 was using 64KB uniform memory while IBM PC
16470                              <1>       ;     compatibles was using 1MB segmented memory
16471                              <1>       ;     in 8086/8088 times.
16472                              <1>       ;
16473                              <1>       ; INPUTS ->
16474                              <1>       ;    u.break - points to end of program
16475                              <1>       ;    u.usp - stack pointer at the moment of swap
16476                              <1>       ;    core - beginning of process program
16477                              <1>       ;    ecore - end of core
16478                              <1>       ;    user - start of user parameter area
16479                              <1>       ;    u.uno - user process number
16480                              <1>       ;    p.dska - holds block number of process
```

```
16481                                  <1>       ; OUTPUTS ->
16482                                  <1>       ;    swp I/O queue
16483                                  <1>       ;    p.break - negative word count of process
16484                                  <1>       ;    r1 - process disk address
16485                                  <1>       ;    r2 - negative word count
16486                                  <1>       ;
16487                                  <1>       ; RETRO UNIX 8086 v1 input/output:
16488                                  <1>       ;
16489                                  <1>       ; INPUTS ->
16490                                  <1>       ;    u.uno - process number (to be swapped out)
16491                                  <1>       ; OUTPUTS ->
16492                                  <1>       ;    none
16493                                  <1>       ;
16494                                  <1>       ;   ((Modified registers: ECX, ESI, EDI))
16495                                  <1>       ;
16496                                  <1>
16497                                  <1>       ; 28/02/2017
16498                                  <1>       ;cmp   byte [multi_tasking], 0  ; Musti tasking mode ?
16499                                  <1>       ;jna   short wswp
16500 000113F6 803D[DA030300]00       <1>       cmp    byte [u.fpsave], 0 ; 28/02/2017
16501 000113FD 7606                   <1>       jna    short wswp
16502 000113FF DD35[DC030300]         <1>       fnsave [u.fpregs] ; save floating point registers (94 bytes)
16503                                  <1> wswp:
16504 00011405 8B3D[B4030300]         <1>       mov    edi, [u.upage] ; process's user (u) structure page addr
16505 0001140B B938000000             <1>       mov    ecx, (U_SIZE + 3) / 4
16506 00011410 BE[5C030300]           <1>       mov    esi, user ; active user (u) structure
16507 00011415 F3A5                   <1>       rep    movsd
16508                                  <1>       ;
16509 00011417 8B35[60030300]         <1>       mov    esi, [u.usp] ; esp (system stack pointer,
16510                                  <1>                           ;      points to user registers)
16511 0001141D 8B0D[5C030300]         <1>       mov    ecx, [u.sp]  ; return address from the system call
16512                                  <1>                           ; (for IRET)
16513                                  <1>                           ; [u.sp] -> EIP (user)
16514                                  <1>                           ; [u.sp+4]-> CS (user)
16515                                  <1>                           ; [u.sp+8] -> EFLAGS (user)
16516                                  <1>                           ; [u.sp+12] -> ESP (user)
16517                                  <1>                           ; [u.sp+16] -> SS (user)
16518 00011423 29F1                   <1>       sub    ecx, esi     ; required space for user registers
16519 00011425 83C114                 <1>       add    ecx, 20           ; +5 dwords to return from system call
16520                                  <1>                           ; (for IRET)
16521 00011428 C1E902                 <1>       shr    ecx, 2
16522 0001142B F3A5                   <1>       rep    movsd
16523 0001142D C3                     <1>       retn
16524                                  <1>
16525                                  <1> rswap:  ; < swap in, swap from disk >
16526                                  <1>       ; 28/02/2017 (frstor)
16527                                  <1>       ; 15/01/2017
16528                                  <1>       ; 14/01/2017
16529                                  <1>       ; 21/05/2016
16530                                  <1>       ; 03/05/2016
16531                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
16532                                  <1>       ; 09/05/2015 - 15/09/2015 (Retro UNIX 386 v1)
16533                                  <1>       ; 26/05/2013 - 08/03/2014 (Retro UNIX 8086 v1)
16534                                  <1>       ; 'rswap' reads a process whose number is in r1,
16535                                  <1>       ; from disk into core.
16536                                  <1>       ;
16537                                  <1>       ; Retro UNIX 386 v1 modification ->
16538                                  <1>       ;       User (u) structure content and the user's register content
16539                                  <1>       ;     will be restored from process's/user's UPAGE (a page for
16540                                  <1>       ;     saving 'u' structure and user registers for task switching).
16541                                  <1>       ;     u.usp - points to kernel stack address which contains
16542                                  <1>       ;             user's registers while entering system call.
16543                                  <1>       ;     u.sp  - points to kernel stack address
16544                                  <1>       ;             to return from system call -for IRET-.
16545                                  <1>       ;     [u.usp]+32+16 = [u.sp]
16546                                  <1>       ;     [u.usp] -> edi, esi, ebp, esp (= [u.usp]+32), ebx,
16547                                  <1>       ;             edx, ecx, eax, gs, fs, es, ds, -> [u.sp].
16548                                  <1>       ;
16549                                  <1>       ; RETRO UNIX 8086 v1 modification ->
16550                                  <1>       ;       'swap to disk' is replaced with 'change running segment'
16551                                  <1>       ;     according to 8086 cpu (x86 real mode) architecture.
16552                                  <1>       ;     pdp-11 was using 64KB uniform memory while IBM PC
16553                                  <1>       ;      compatibles was using 1MB segmented memory
16554                                  <1>       ;      in 8086/8088 times.
16555                                  <1>       ;
16556                                  <1>       ; INPUTS ->
16557                                  <1>       ;    r1 - process number of process to be read in
16558                                  <1>       ;    p.break - negative of word count of process
16559                                  <1>       ;    p.dska - disk address of the process
16560                                  <1>       ;    u.emt - determines handling of emt's
16561                                  <1>       ;    u.ilgins - determines handling of illegal instructions
16562                                  <1>       ; OUTPUTS ->
16563                                  <1>       ;    8 = (u.ilgins)
16564                                  <1>       ;    24 = (u.emt)
16565                                  <1>       ;    swp - bit 10 is set to indicate read
16566                                  <1>       ;             (bit 15=0 when reading is done)
16567                                  <1>       ;    swp+2 - disk block address
16568                                  <1>       ;    swp+4 - negative word count
16569                                  <1>       ;      ((swp+6 - address of user structure))
16570                                  <1>       ;
16571                                  <1>       ; RETRO UNIX 8086 v1 input/output:
16572                                  <1>       ;
16573                                  <1>       ; INPUTS ->
16574                                  <1>       ;    AL      - new process number (to be swapped in)
16575                                  <1>       ; OUTPUTS ->
16576                                  <1>       ;    none
16577                                  <1>       ;
16578                                  <1>       ;   ((Modified registers: EAX, ECX, ESI, EDI, ESP))
16579                                  <1>       ;
16580                                  <1>       ; Retro UNIX 386 v1 - modification ! 14/05/2015
16581 0001142E 89C6                   <1>       mov    esi, eax  ; process's user (u) structure page addr
16582 00011430 B938000000             <1>       mov    ecx, (U_SIZE + 3) / 4
16583 00011435 BF[5C030300]           <1>       mov    edi, user ; active user (u) structure
16584 0001143A F3A5                   <1>       rep    movsd
16585 0001143C 58                     <1>       pop    eax    ; 'rswap' return address
```

```
16586                                  <1>       ;
16587                                  <1>       ;cli
16588 0001143D 8B3D[60030300]          <1>       mov    edi, [u.usp] ; esp (system stack pointer,
16589                                  <1>                           ;     points to user registers)
16590 00011443 89FC                    <1>       mov    esp, edi     ; 14/01/2017
16591 00011445 8B0D[5C030300]          <1>       mov    ecx, [u.sp]  ; return address from the system call
16592                                  <1>                           ; (for IRET)
16593                                  <1>                           ; [u.sp] -> EIP (user)
16594                                  <1>                           ; [u.sp+4]-> CS (user)
16595                                  <1>                           ; [u.sp+8] -> EFLAGS (user)
16596                                  <1>                           ; [u.sp+12] -> ESP (user)
16597                                  <1>                           ; [u.sp+16] -> SS (user)
16598 0001144B 29F9                    <1>       sub    ecx, edi     ; required space for user registers
16599 0001144D 83C114                  <1>       add    ecx, 20           ; +5 dwords to return from system call
16600                                  <1>                           ; (for IRET)
16601 00011450 C1E902                  <1>       shr    ecx, 2
16602 00011453 F3A5                    <1>       rep    movsd
16603                                  <1>       ;mov   esp, [u.usp] ; 15/09/2015
16604                                  <1>       ;sti
16605                                  <1>       ; 28/02/2017
16606                                  <1>       ;cmp   byte [multi_tasking], 0  ; Musti tasking mode ?
16607                                  <1>       ;jna   short rswp_retn
16608 00011455 803D[DA030300]00        <1>       cmp    byte [u.fpsave], 0
16609 0001145C 7606                    <1>       jna    short rswp_retn
16610 0001145E DD25[DC030300]          <1>       frstor [u.fpregs] ; restore floating point regs (94 bytes)
16611                                  <1> rswp_retn:
16612 00011464 50                      <1>       push   eax    ; 'rswap' return address
16613 00011465 C3                      <1>       retn
16614                                  <1>
16615                                  <1> putlu:
16616                                  <1>       ; 20/05/2016
16617                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
16618                                  <1>       ; 10/05/2015 - 12/09/2015 (Retro UNIX 386 v1)
16619                                  <1>       ; 15/04/2013 - 23/02/2014 (Retro UNIX 8086 v1)
16620                                  <1>       ; 'putlu' is called with a process number in r1 and a pointer
16621                                  <1>       ; to lowest priority Q (runq+4) in r2. A link is created from
16622                                  <1>       ; the last process on the queue to process in r1 by putting
16623                                  <1>       ; the process number in r1 into the last process's link.
16624                                  <1>       ;
16625                                  <1>       ; INPUTS ->
16626                                  <1>       ;    r1 - user process number
16627                                  <1>       ;    r2 - points to lowest priority queue
16628                                  <1>       ;    p.dska - disk address of the process
16629                                  <1>       ;    u.emt - determines handling of emt's
16630                                  <1>       ;    u.ilgins - determines handling of illegal instructions
16631                                  <1>       ; OUTPUTS ->
16632                                  <1>       ;    r3 - process number of last process on the queue upon
16633                                  <1>       ;       entering putlu
16634                                  <1>       ;    p.link-1 + r3 - process number in r1
16635                                  <1>       ;    r2 - points to lowest priority queue
16636                                  <1>       ;
16637                                  <1>       ; ((Modified registers: EDX, EBX))
16638                                  <1>       ;
16639                                  <1>       ; / r1 = user process no.; r2 points to lowest priority queue
16640                                  <1>
16641                                  <1>       ; EBX = r2
16642                                  <1>       ; EAX = r1 (AL=r1b)
16643                                  <1>
16644                                  <1>       ; 20/05/2016
16645                                  <1>       ; AL = process number (1 to 16) // Retro UNIX 8086, 386 v1 //
16646                                  <1>       ;     (max. 16 processes available for current kernel version)
16647                                  <1>       ; EBX = run queue address ; 20/05/2016 (TRDOS 386)
16648                                  <1>             ; which is one of following addresses:
16649                                  <1>             ;  1) 'runq_event' high priority run queue
16650                                  <1>             ;  2) 'runq_normal' normal/regular priority run queue
16651                                  <1>             ;  3) 'runq_background' low priority run queue
16652                                  <1>
16653                                  <1>       ;mov   ebx, runq
16654 00011466 0FB613                  <1>       movzx  edx, byte [ebx]
16655 00011469 43                      <1>       inc    ebx
16656 0001146A 20D2                    <1>       and    dl, dl
16657                                  <1>             ; tstb (r2)+ / is queue empty?
16658 0001146C 740A                    <1>             jz     short putlu_1
16659                                  <1>             ; beq 1f / yes, branch
16660 0001146E 8A13                    <1>       mov    dl, [ebx] ; 12/09/2015
16661                                  <1>             ; movb (r2),r3 / no, save the "last user" process number
16662                                  <1>                         ; / in r3
16663 00011470 8882[9F000300]          <1>       mov    [edx+p.link-1], al
16664                                  <1>             ; movb r1,p.link-1(r3) / put pointer to user on
16665                                  <1>                         ; / "last users" link
16666 00011476 EB03                    <1>       jmp    short putlu_2
16667                                  <1>             ; br 2f /
16668                                  <1> putlu_1: ; 1:
16669 00011478 8843FF                  <1>       mov    [ebx-1], al
16670                                  <1>                    ; movb r1,-1(r2) / user is only user;
16671                                  <1>                        ; / put process no. at beginning and at end
16672                                  <1> putlu_2: ; 2:
16673 0001147B 8803                    <1>       mov    [ebx], al
16674                                  <1>                    ; movb r1,(r2) / user process in r1 is now the last entry
16675                                  <1>                        ; / on the queue
16676 0001147D 88C2                    <1>       mov    dl, al
16677 0001147F 88B2[9F000300]          <1>       mov    [edx+p.link-1], dh ; 0
16678                                  <1>             ; dec r2 / restore r2
16679 00011485 C3                      <1>       retn
16680                                  <1>             ; rts r0
16681                                  <1>
16682                                  <1> sysver:
16683                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
16684 00011486 C705[64030300]0002-     <1>       mov    dword [u.r0], 200h ; AH = major version, AL = minor version
16684 0001148E 0000                    <1>
16685 00011490 E93BBFFFFF              <1>       jmp    sysret
16686                                  <1>
16687                                  <1>
16688                                  <1> syspri: ; change running priority (of the process)
16689                                  <1>       ; 21/05/2016
```

```
16690                                 <1>       ; 20/05/2026 - TRDOS 386 (TRDOS v2.0)
16691                                 <1>       ; INPUT ->
16692                                 <1>       ;     BL = priority level
16693                                 <1>       ;        0 = low running priority (running on background)
16694                                 <1>       ;        1 = normal/regular priority (running as regular)
16695                                 <1>       ;        2 = high/event priority (running for event)
16696                                 <1>       ;        >2 = invalid, it will accepted as 2 (event)
16697                                 <1>       ;        0FFh = get/return current running priority only
16698                                 <1>       ; OUTPUT ->
16699                                 <1>       ;     * if current [u.pri] < 2
16700                                 <1>       ;       if BL input < 0FFh ->
16701                                 <1>       ;          [u.pri] is updated as in BL input (0,1,2)
16702                                 <1>       ;       if BL input = 0FFh -> AL = [u.pri] (current)
16703                                 <1>       ;
16704                                 <1>       ;     * if current [u.pri] = 2
16705                                 <1>       ;       if BL input < 0FFh -> cf = 1 & AL = 2
16706                                 <1>       ;       if BL input = 0FFh -> cf = 0 & AL = 2
16707                                 <1>       ;
16708                                 <1>       ;     NOTE:
16709                                 <1>       ;     If [u.pri] = 2, it can not be changed to 1 or 0;
16710                                 <1>       ;     because, run queue of the running process is unspecified
16711                                 <1>       ;     at this      stage. Process might be started by a timer event
16712                                 <1>       ;     or priority might be changed to high by previous
16713                                 <1>       ;     'syspri' system     call. In both cases, the process is in
16714                                 <1>       ;     'runq_normal' or 'runq_background' queue.
16715                                 <1>       ;     As result of this fact, when the [u.quant] time quantum
16716                                 <1>       ;     of the process is elapsed or 'sysrele' system call is
16717                                 <1>       ;     instructed by the process, 'tswap' ('tswitch') procedure
16718                                 <1>       ;     will be called (to 'swap' or 'switch' out the procedure)
16719                                 <1>       ;     and it will not call 'putlu' to add the (stopping)
16720                                 <1>       ;     process to relevant run queue when [u.pri] = 2.
16721                                 <1>       ;     (Otherwise, it would be possible to add process to
16722                                 <1>       ;     a run queue while it is already in a run queue, wrongly.)
16723                                 <1>       ;
16724                                 <1>       ;     If [u.pri]< 2, 'tswap/tswitch' procedure will call
16725                                 <1>       ;     'putlu' to add process to relevant run queue
16726                                 <1>       ;     according to [u.pri] value. ('runq_normal' for 1,
16727                                 <1>       ;     'runq_background' for 0).
16728                                 <1>       ;
16729                                 <1>       ;     If BL input >= 2 and < 0FFh while [u.pri] < 2,
16730                                 <1>       ;     process will be added to 'runq_normal' queue and
16731                                 <1>       ;     [u.pri] will be set to 2. (in 'syspri' system call)
16732                                 <1>       ;
16733                                 <1>
16734 00011495 29C0                  <1>       sub   eax, eax ; 0
16735 00011497 A3[C8030300]          <1>       mov   [u.error], eax
16736                                 <1>
16737 0001149C A0[A9030300]          <1>       mov   al, [u.pri]
16738 000114A1 A3[64030300]          <1>       mov   [u.r0], eax
16739                                 <1>
16740 000114A6 FEC3                  <1>       inc   bl
16741 000114A8 0F8422BFFFFF          <1>       jz    sysret ; 0FFh -> 0, get priority level
16742                                 <1>
16743 000114AE 3C02                  <1>       cmp   al, 2
16744 000114B0 0F83FABEFFFF          <1>       jnb   error ; CF = 1 & AL = 2 (& last error = 0)
16745                                 <1>
16746 000114B6 FECB                  <1>       dec   bl
16747 000114B8 80FB02                <1>       cmp   bl, 2
16748 000114BB 7602                  <1>       jna   short syspri_1
16749 000114BD B302                  <1>       mov   bl, 2
16750                                 <1> syspri_1:
16751 000114BF 881D[A9030300]        <1>       mov   [u.pri], bl
16752 000114C5 80FB02                <1>       cmp   bl, 2
16753 000114C8 0F8202BFFFFF          <1>       jb    sysret
16754                                 <1>
16755                                 <1>       ; here...
16756                                 <1>       ; Priority of current process has been changed to high
16757                                 <1>       ; ('run for event') but current process will be added to
16758                                 <1>       ; 'run as normal' queue. ('run for event' high priority
16759                                 <1>       ; queue is under control of timer -& RTC- interrupt only!)
16760                                 <1>       ;
16761                                 <1>       ; (Otherwise, process can fall into black hole!
16762                                 <1>       ; e.g. if it is not in waiting list and it has not got
16763                                 <1>       ; a timer event and it is not in a run queue!
16764                                 <1>       ; Because, when [u.pri] is 2, 'tswap/tswitch' will not
16765                                 <1>       ; add the stopping process to a run queue.)
16766                                 <1>
16767 000114CE A0[B3030300]          <1>       mov   al, [u.uno]
16768 000114D3 BB[54030300]          <1>       mov   ebx, runq_normal ; normal priority !
16769                                 <1>                              ; [u.pri] is set to high
16770                                 <1>                              ; but 'runq_event' queue is set
16771                                 <1>                              ; only by the kernel's timer
16772                                 <1>                              ; event function (timer interrupt).
16773 000114D8 E889FFFFFF            <1>       call  putlu
16774 000114DD E9EEBEFFFF            <1>       jmp   sysret
16775                                 <1>
16776                                 <1> cpass: ; / get next character from user area of core and put it in AL (r1)
16777                                 <1>       ; 02/05/2016 - TRDOS 386 (TRDOS v2.0)
16778                                 <1>       ; 19/05/2015 - 18/10/2015 (Retro UNIX 386 v1)
16779                                 <1>       ; 14/08/2013 - 20/09/2013 (Retro UNIX 8086 v1)
16780                                 <1>       ; INPUTS ->
16781                                 <1>       ;     [u.base] = virtual address in user area
16782                                 <1>       ;     [u.count] = byte count (max.)
16783                                 <1>       ;     [u.pcount] = byte count in page (0 = reset)
16784                                 <1>       ; OUTPUTS ->
16785                                 <1>       ;     AL = the character which is pointed by [u.base]
16786                                 <1>       ;     zf = 1 -> transfer count has been completed
16787                                 <1>         ;
16788                                 <1>       ; ((Modified registers:  EAX, EDX, ECX))
16789                                 <1>       ;
16790 000114E2 833D[88030300]00      <1>       cmp   dword [u.count], 0  ; have all the characters been transferred
16791                                 <1>                               ; i.e., u.count, # of chars. left
16792 000114E9 763F                  <1>       jna   short cpass_3    ; to be transferred = 0?) yes, branch
16793 000114EB FF0D[88030300]        <1>       dec   dword [u.count]         ; no, decrement u.count
16794                                 <1>         ; 19/05/2015
```

```
16795                                  <1>       ;(Retro UNIX 386 v1 - translation from user's virtual address
16796                                  <1>       ;                    to physical address
16797 000114F1 66833D[C4030300]00      <1>       cmp    word [u.pcount], 0 ; byte count in page = 0 (initial value)
16798                                  <1>                               ; 1-4095 --> use previous physical base address
16799                                  <1>                               ; in [u.pbase]
16800 000114F9 770E                    <1>       ja     short cpass_1
16801 000114FB 833D[BC030300]00        <1>       cmp    dword [u.ppgdir], 0 ; is the caller os kernel
16802 00011502 7427                    <1>       je     short cpass_k       ; (sysexec, '/etc/init') ?  (MainProg)
16803 00011504 E858FDFFFF              <1>       call   trans_addr_r
16804                                  <1> cpass_1:
16805 00011509 66FF0D[C4030300]        <1>       dec    word [u.pcount]
16806                                  <1> cpass_2:
16807 00011510 8B15[C0030300]          <1>       mov    edx, [u.pbase]
16808 00011516 8A02                    <1>       mov    al, [edx]    ; take the character pointed to
16809                                  <1>                            ; by u.base and put it in r1
16810 00011518 FF05[8C030300]          <1>       inc    dword [u.nread] ; increment no. of bytes transferred
16811 0001151E FF05[84030300]          <1>       inc    dword [u.base]  ; increment the buffer address to point to the
16812                                  <1>                              ; next byte
16813 00011524 FF05[C0030300]          <1>       inc    dword [u.pbase]
16814                                  <1> cpass_3:
16815 0001152A C3                      <1>       retn
16816                                  <1> cpass_k:
16817                                  <1>       ; 02/07/2015
16818                                  <1>       ; The caller is os kernel
16819                                  <1>       ; (get sysexec arguments from kernel's memory space)
16820 0001152B 8B1D[84030300]          <1>       mov    ebx, [u.base]
16821 00011531 66C705[C4030300]00-     <1>       mov    word [u.pcount], PAGE_SIZE ; 4096
16821 00011539 10                      <1>
16822 0001153A 891D[C0030300]          <1>       mov    [u.pbase], ebx
16823 00011540 EBCE                    <1>       jmp    short cpass_2
16824                                  <1>
16825                                  <1> transfer_to_user_buffer: ; fast transfer
16826                                  <1>       ; 27/05/2016
16827                                  <1>       ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
16828                                  <1>       ;
16829                                  <1>       ; INPUT ->
16830                                  <1>       ;      ESI = source address in system space
16831                                  <1>       ;      EDI = user's buffer address
16832                                  <1>       ;      ECX = transfer (byte) count
16833                                  <1>       ;      [u.pgdir] = user's page directory
16834                                  <1>       ; OUTPUT ->
16835                                  <1>       ;      ECX = actual transfer count
16836                                  <1>       ;      cf = 1 -> error
16837                                  <1>       ;      [u.count] = remain byte count
16838                                  <1>       ;
16839                                  <1>       ; Modified registers: eax, ecx
16840                                  <1>       ;
16841                                  <1>
16842 00011542 21C9                    <1>       and    ecx, ecx
16843 00011544 743B                    <1>       jz     short ttub_4
16844                                  <1>
16845 00011546 890D[88030300]          <1>       mov    [u.count], ecx
16846                                  <1>
16847 0001154C 57                      <1>       push   edi
16848 0001154D 56                      <1>       push   esi
16849 0001154E 53                      <1>       push   ebx
16850 0001154F 52                      <1>       push   edx
16851 00011550 51                      <1>       push   ecx
16852                                  <1>
16853 00011551 89FB                    <1>       mov    ebx, edi
16854 00011553 81C300004000            <1>       add    ebx, CORE ; 27/05/2016
16855                                  <1> ttub_1:
16856                                  <1>       ; ebx = virtual (linear) address
16857                                  <1>       ; [u.pgdir] = user's page directory
16858 00011559 E80A49FFFF              <1>             call   get_physical_addr_x ; get physical address
16859 0001155E 7222                    <1>       jc     short ttub_5
16860                                  <1>       ; eax = physical address
16861                                  <1>       ; ecx = remain byte count in page (1-4096)
16862 00011560 89C7                    <1>       mov    edi, eax
16863 00011562 A1[88030300]            <1>       mov    eax, [u.count]
16864 00011567 39C1                    <1>       cmp    ecx, eax
16865 00011569 7602                    <1>       jna    short ttub_2
16866 0001156B 89C1                    <1>       mov    ecx, eax
16867                                  <1> ttub_2:
16868 0001156D 29C8                    <1>       sub    eax, ecx
16869 0001156F 01CB                    <1>       add    ebx, ecx
16870 00011571 F3A4                    <1>       rep    movsb
16871 00011573 A3[88030300]            <1>       mov    [u.count], eax
16872 00011578 09C0                    <1>       or     eax, eax
16873 0001157A 75DD                    <1>       jnz    short ttub_1
16874                                  <1> ttub_retn:
16875                                  <1> tfub_retn:
16876 0001157C 59                      <1>       pop    ecx ; transfer count = actual transfer count
16877                                  <1> ttub_3:
16878 0001157D 5A                      <1>       pop    edx
16879 0001157E 5B                      <1>       pop    ebx
16880 0001157F 5E                      <1>       pop    esi
16881 00011580 5F                      <1>       pop    edi
16882                                  <1> ttub_4:
16883 00011581 C3                      <1>       retn
16884                                  <1> ttub_5:
16885 00011582 59                      <1>       pop    ecx
16886 00011583 2B0D[88030300]          <1>       sub    ecx, [u.count] ; actual transfer count
16887 00011589 F9                      <1>       stc
16888 0001158A EBF1                    <1>       jmp    short ttub_3
16889                                  <1>
16890                                  <1> transfer_from_user_buffer: ; fast transfer
16891                                  <1>       ; 27/05/2016
16892                                  <1>       ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
16893                                  <1>       ;
16894                                  <1>       ; INPUT ->
16895                                  <1>       ;      ESI = user's buffer address
16896                                  <1>       ;      EDI = destination address in system space
16897                                  <1>       ;      ECX = transfer (byte) count
16898                                  <1>       ;      [u.pgdir] = user's page directory
```

```
16899                                 <1>       ; OUTPUT ->
16900                                 <1>       ;      ecx = actual transfer count
16901                                 <1>       ;      cf = 1 -> error
16902                                 <1>       ;      [u.count] = remain byte count
16903                                 <1>       ;
16904                                 <1>       ; Modified registers: eax, ecx
16905                                 <1>       ;
16906                                 <1>
16907 0001158C 21C9                  <1>       and    ecx, ecx
16908                                 <1>       ;jz    short tfub_4
16909 0001158E 74F1                  <1>       jz     short ttub_4
16910                                 <1>
16911 00011590 890D[88030300]        <1>       mov    [u.count], ecx
16912                                 <1>
16913 00011596 57                    <1>       push   edi
16914 00011597 56                    <1>       push   esi
16915 00011598 53                    <1>       push   ebx
16916 00011599 52                    <1>       push   edx
16917 0001159A 51                    <1>       push   ecx
16918                                 <1>
16919 0001159B 89F3                  <1>       mov    ebx, esi
16920 0001159D 81C300004000          <1>       add    ebx, CORE ; 27/05/2016
16921                                 <1> tfub_1:
16922                                 <1>       ; ebx = virtual (linear) address
16923                                 <1>       ; [u.pgdir] = user's page directory
16924 000115A3 E8C048FFFF            <1>             call   get_physical_addr_x ; get physical address
16925                                 <1>       ;jc    short tfub_5
16926 000115A8 72D8                  <1>       jc     short ttub_5
16927                                 <1>       ; eax = physical address
16928                                 <1>       ; ecx = remain byte count in page (1-4096)
16929 000115AA 89C6                  <1>       mov    esi, eax
16930 000115AC A1[88030300]          <1>       mov    eax, [u.count]
16931 000115B1 39C1                  <1>       cmp    ecx, eax
16932 000115B3 7602                  <1>       jna    short tfub_2
16933 000115B5 89C1                  <1>       mov    ecx, eax
16934                                 <1> tfub_2:
16935 000115B7 29C8                  <1>       sub    eax, ecx
16936 000115B9 01CB                  <1>       add    ebx, ecx
16937 000115BB F3A4                  <1>       rep    movsb
16938 000115BD A3[88030300]          <1>       mov    [u.count], eax
16939 000115C2 09C0                  <1>       or     eax, eax
16940 000115C4 75DD                  <1>       jnz    short tfub_1
16941                                 <1>
16942 000115C6 EBB4                  <1>       jmp    short tfub_retn
16943                                 <1>
16944                                 <1> ;tfub_retn:
16945                                 <1> ;     pop    ecx ; transfer count = actual transfer count
16946                                 <1> ;tfub_3:
16947                                 <1> ;     pop    edx
16948                                 <1> ;     pop    ebx
16949                                 <1> ;     pop    esi
16950                                 <1> ;     pop    edi
16951                                 <1> ;tfub_4:
16952                                 <1> ;     retn
16953                                 <1> ;tfub_5:
16954                                 <1> ;     pop    ecx
16955                                 <1> ;     sub    ecx, [u.count] ; actual transfer count
16956                                 <1> ;     stc
16957                                 <1> ;     jmp    short tfub_3
16958                                 <1>
16959                                 <1> sysfff: ; <Find First File>
16960                                 <1>       ; 17/10/2016
16961                                 <1>       ; 16/10/2016
16962                                 <1>       ; 15/10/2016 TRDOS 386 (TRDOS v2.0) feature only !
16963                                 <1>       ;           -derived from TRDOS v1.0, INT_21H.ASM-
16964                                 <1>       ;            ("loc_INT21h_find_first_file")
16965                                 <1>       ; TRDOS 8086 (v1.0)
16966                                 <1>       ;      07/08/2011
16967                                 <1>       ;      Find First File
16968                                 <1>       ;      INPUT:
16969                                 <1>       ;          CX= Attributes
16970                                 <1>       ;          DS:DX= Pointer to filename
16971                                 <1>       ;      MSDOS OUTPUT:
16972                                 <1>       ;          DTA: (Default address: PSP offset 80h)
16973                                 <1>       ;          Offset  Descrription
16974                                 <1>       ;          0      Reserved for use find next file
16975                                 <1>       ;          21     Attribute of file found
16976                                 <1>       ;          22     Time stamp of file
16977                                 <1>       ;          24     Date stamp of file
16978                                 <1>       ;          26     File size in bytes
16979                                 <1>       ;          30     Filename and extension (zero terminated)
16980                                 <1>       ;      If cf = 1:
16981                                 <1>       ;          Error Codes: (in AX)
16982                                 <1>       ;             2 - File not found
16983                                 <1>       ;             18 - No more files
16984                                 <1>       ;
16985                                 <1>       ; TRDOS 386 (v2.0)
16986                                 <1>       ; 15/10/2016
16987                                 <1>       ;
16988                                 <1>       ; INPUT ->
16989                                 <1>       ;        CL = File attributes
16990                                 <1>       ;           bit 0 (1) - Read only file (R)
16991                                 <1>       ;            bit 1 (1) - Hidden file (H)
16992                                 <1>       ;            bit 2 (1) - System file (R)
16993                                 <1>       ;            bit 3 (1) - Volume label/name (V)
16994                                 <1>       ;            bit 4 (1) - Subdirectory (D)
16995                                 <1>       ;           bit 5 (1) - File has been archived (A)
16996                                 <1>       ;        CH = 0 -> Return basic parameters (24 bytes)
16997                                 <1>       ;        CH > 0 -> Return FindFile structure/table (128 bytes)
16998                                 <1>       ;         EBX = Pointer to filename (ASCIIZ) -path-
16999                                 <1>       ;        EDX = File parameters buffer address
17000                                 <1>       ;            (buffer size = 24 bytes if CH input = 0)
17001                                 <1>       ;            (buffer size = 128 bytes if CH input > 0)
17002                                 <1>       ;
17003                                 <1>       ; OUTPUT ->
```

```
17004                                  <1>       ;         EAX = 0 if CH input > 0
17005                                  <1>       ;         EAX = First cluster number of file if CH input = 0
17006                                  <1>       ;         EDX = File parameters table/structure address
17007                                  <1>       ;         Basic Parameters:
17008                                  <1>       ;             Offset  Description
17009                                  <1>       ;             ------ ---------------
17010                                  <1>       ;             0      File Attributes
17011                                  <1>       ;             1      Ambiguous filename chars are used sign
17012                                  <1>       ;                    (0 = filename fits exactly with request)
17013                                  <1>       ;                    (>0 = ambiguous filename chars are used)
17014                                  <1>       ;             2      Time stamp of file
17015                                  <1>       ;             4      Date stamp of file
17016                                  <1>       ;             6      File size in bytes
17017                                  <1>       ;             10     Short Filename (ASCIIZ, max. 13 bytes)
17018                                  <1>       ;             23     Longname Length (1-255) if existing
17019                                  <1>       ;
17020                                  <1>       ;           cf = 1 -> Error code in AL
17021                                  <1>       ;
17022                                  <1>       ; Modified Registers: EAX (at the return of system call)
17023                                  <1>       ;
17024                                  <1>       ; TR-DOS FindFile (FFF) Structure (128 bytes):
17025                                  <1>       ; 09/10/2011 (DIR.ASM) - 10/02/2016 (trdoskx.s)
17026                                  <1>       ;
17027                                  <1>       ; Offset     Parameter           Size
17028                                  <1>       ; ------     ------------------  --------
17029                                  <1>       ; 0          FindFile_Drv        1 byte
17030                                  <1>       ; 1          FindFile_Directory  65 bytes
17031                                  <1>       ; 66         FindFile_Name       13 bytes
17032                                  <1>       ; 79         FindFile_LongNameEntryLength 1 byte
17033                                  <1>       ;Above 80 bytes form
17034                                  <1>       ;TR-DOS Source/Destination File FullName Format/Structure
17035                                  <1>       ; 80         FindFile_AttributesMask 1 word
17036                                  <1>       ; 82         FindFile_DirEntry   32 bytes (*)
17037                                  <1>       ; 114        FindFile_DirFirstCluster 1 double word
17038                                  <1>       ; 118        FindFile_DirCluster 1 double word
17039                                  <1>       ; 122        FindFile_DirEntryNumber 1 word
17040                                  <1>       ; 124        FindFile_MatchCounter     1 word
17041                                  <1>       ; 126        FindFile_Reserved   1 word
17042                                  <1>       ; (*) MS-DOS, FAT 12-16-32 classic directory entry (32 bytes)
17043                                  <1>
17044                                  <1>       ;mov   [u.namep], ebx
17045                                  <1>       ; 16/10/2016
17046 000115C8 8915[588D0100]          <1>       mov    [FFF_UBuffer], edx
17047 000115CE 66890D[5D8D0100]        <1>       mov    [FFF_Attrib], cx ; [FFF_RType] = ch
17048                                  <1>                   ; Attributes in CL, return data type in CH
17049 000115D5 89DE                    <1>       mov    esi, ebx
17050                                  <1>       ; file name is forced, change directory as temporary
17051                                  <1>       ;mov   ax, 1
17052                                  <1>       ;mov   [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
17053                                  <1>       ;call  set_working_path
17054 000115D7 E8E2130000              <1>       call   set_working_path_x ; 17/10/2016
17055 000115DC 731D                    <1>       jnc    short sysfff_0
17056                                  <1>
17057 000115DE 21C0                    <1>       and    eax, eax  ; 0 -> Bad Path!
17058 000115E0 7505                    <1>       jnz    short sysfff_err
17059                                  <1>
17060                                  <1>       ; eax = 0
17061 000115E2 B80C000000              <1>       mov    eax, ERR_DIR_NOT_FOUND ; Directory not found !
17062                                  <1> sysfff_err:
17063 000115E7 A3[64030300]            <1>       mov    [u.r0], eax
17064 000115EC A3[C8030300]            <1>       mov    [u.error], eax
17065 000115F1 E89D140000              <1>       call   reset_working_path
17066 000115F6 E9B5BDFFFF              <1>       jmp    error
17067                                  <1>
17068                                  <1> sysfff_0:
17069                                  <1>       ;sub   ah, ah ; ah = 0
17070 000115FB 8A0424                  <1>       mov    al, [esp]
17071 000115FE 08C0                    <1>       or     al, al
17072 00011600 7412                    <1>       jz     short sysfff_2
17073 00011602 B410                    <1>       mov    ah, 10h
17074 00011604 A808                    <1>       test   al, 08h
17075 00011606 7503                    <1>       jnz    short sysfff_1
17076 00011608 80CC08                  <1>       or     ah, 08h
17077                                  <1> sysfff_1:
17078 0001160B 2410                    <1>       and    al, 10h ; Directory
17079 0001160D 7405                    <1>       jz     short sysfff_2
17080 0001160F 80E408                  <1>       and    ah, 08h
17081 00011612 30C0                    <1>       xor    al, al ; When a directory is searched,
17082                                  <1>                     ; filename will be returned even if
17083                                  <1>                     ; it is not a directory!
17084                                  <1>                     ; Because: (in order to prevent
17085                                  <1>                     ; creating a dir with existing file name)
17086                                  <1>                     ; Dir and file names must not be same!
17087                                  <1>                     ; (return attribute must be checked)
17088                                  <1> sysfff_2:
17089                                  <1>       ; AX = Attributes mask
17090                                  <1>             ; AL = AND mask (result must be equal to AL)
17091                                  <1>             ; AH = Negative AND mask (result must be ZERO)
17092                                  <1>       ; ESI = FindFile_Name address
17093                                  <1>
17094 00011614 E83C79FFFF              <1>       call   find_first_file
17095 00011619 72CC                    <1>       jc     short sysfff_err ; eax = 2 (File not found !)
17096                                  <1>
17097                                  <1>       ; ESI = Directory Entry (FindFile_DirEntry) Location
17098                                  <1>       ; EDI = Directory Buffer Directory Entry Location
17099                                  <1>       ; EAX = File Size
17100                                  <1>       ;  BL = Attributes of The File/Directory
17101                                  <1>       ;  BH = Long Name Yes/No Status (>0 is YES)
17102                                  <1>       ;  DX > 0 : Ambiguous filename chars are used
17103                                  <1>
17104                                  <1> sysfff_3:
17105                                  <1>       ; 16/10/2016
17106 0001161B 668B0D[5D8D0100]        <1>       mov    cx, [FFF_Attrib]
17107                                  <1>       ; Attribs in CL, return data type in CH
17108                                  <1>
```

```
17109                                  <1>       ;or    cl, cl
17110                                  <1>       ;jz    short sysfff_4 ; 0 = No filter
17111 00011622 80F1FF                  <1>       xor    cl, 0FFh
17112 00011625 20D9                    <1>       and    cl, bl
17113 00011627 7409                    <1>       jz     short sysfff_4
17114                                  <1>
17115                                  <1>       ;mov   eax, 2 ; 'file not found !' error
17116                                  <1>       ;jmp   short sysfff_err_1
17117                                  <1>
17118                                  <1>       ; 16/10/2016
17119 00011629 E8D679FFFF              <1>       call   find_next_file
17120 0001162E 72B7                    <1>       jc     short sysfff_err ; eax = 12 (no more files !)
17121 00011630 EBE9                    <1>       jmp    short sysfff_3
17122                                  <1>
17123                                  <1> sysfff_4:
17124 00011632 20ED                    <1>       and    ch, ch ; [FFF_RType]
17125 00011634 7412                    <1>       jz     short sysfff_5
17126 00011636 B980000000              <1>       mov    ecx, 128 ; ; transfer length
17127 0001163B 880D[5C8D0100]          <1>       mov    [FFF_Valid], cl
17128                                  <1> sysfnf_11:
17129 00011641 BE[0E8A0100]            <1>       mov    esi, FindFile_Drv
17130 00011646 EB44                    <1>       jmp    short sysfff_6
17131                                  <1> sysfff_5:
17132                                  <1>       ;mov   esi, FindFile_DirEntry
17133 00011648 B918000000              <1>       mov    ecx, 24  ; transfer length
17134 0001164D 880D[5C8D0100]          <1>       mov    [FFF_Valid], cl
17135                                  <1> sysfnf_12:
17136 00011653 BF[E48E0100]            <1>       mov    edi, DTA ; FFF data transfer address
17137                                  <1>       ;mov   al, [esi+DirEntry_Attr] ; 11
17138 00011658 88D8                    <1>       mov    al, bl ; File/Dir Attributes
17139 0001165A 887F17                  <1>       mov    [edi+23], bh ; Longname length (0= none)
17140 0001165D AA                      <1>       stosb
17141 0001165E 88D0                    <1>       mov    al, dl ; DL is for '?'
17142 00011660 00F0                    <1>       add    al, dh ; DH is for '*'
17143                                  <1>       ; AL > 0 if ambiguous file name wildcards are used
17144 00011662 AA                      <1>       stosb
17145 00011663 8B4616                  <1>       mov    eax, [esi+DirEntry_WrtTime] ; 22
17146 00011666 AB                      <1>       stosd ; DirEntry_WrtTime & DirEntry_WrtDate
17147 00011667 8B461C                  <1>       mov    eax, [esi+DirEntry_FileSize] ; 28
17148 0001166A AB                      <1>       stosd
17149 0001166B 668B4614                <1>       mov    ax, [esi+DirEntry_FstClusHI] ; 20
17150 0001166F 66C1E010                <1>       shl    ax, 16
17151 00011673 668B461A                <1>       mov    ax, [esi+DirEntry_FstClusLO] ; 26
17152 00011677 A3[64030300]            <1>       mov    [u.r0], eax ; First Cluster
17153                                  <1>
17154                                  <1>       ;mov   esi, FindFile_DirEntry
17155 0001167C E84F140000              <1>       call   get_file_name
17156                                  <1>
17157 00011681 8A0D[5C8D0100]          <1>       mov    cl, [FFF_Valid]
17158 00011687 BE[E48E0100]            <1>       mov    esi, DTA ; FFF data transfer address
17159                                  <1> sysfff_6:
17160 0001168C 8B3D[588D0100]          <1>       mov    edi, [FFF_UBuffer] ; user's buffer address (edx)
17161 00011692 E8ABFEFFFF              <1>       call   transfer_to_user_buffer
17162                                  <1>
17163 00011697 890D[64030300]          <1>       mov    [u.r0], ecx ; actual transfer count
17164 0001169D E8F1130000              <1>       call   reset_working_path
17165 000116A2 E929BDFFFF              <1>       jmp    sysret
17166                                  <1>
17167                                  <1> sysfnf: ; <Find Next File>
17168                                  <1>       ; 16/10/2016 TRDOS 386 (TRDOS v2.0) feature only !
17169                                  <1>       ;           -derived from TRDOS v1.0, INT_21H.ASM-
17170                                  <1>       ;            ("loc_INT21h_find_next_file")
17171                                  <1>       ; TRDOS 8086 (v1.0)
17172                                  <1>         ;   07/08/2011
17173                                  <1>         ;   Find First File
17174                                  <1>       ;     INPUT:
17175                                  <1>         ;       none
17176                                  <1>       ;     MSDOS OUTPUT:
17177                                  <1>       ;         DTA: (Default address: PSP offset 80h)
17178                                  <1>       ;         Offset  Descrription
17179                                  <1>       ;         0     Reserved for use find next file
17180                                  <1>       ;         21    Attribute of file found
17181                                  <1>       ;         22    Time stamp of file
17182                                  <1>       ;         24    Date stamp of file
17183                                  <1>       ;         26    File size in bytes
17184                                  <1>       ;         30    Filename and extension (zero terminated)
17185                                  <1>       ;     If cf = 1:
17186                                  <1>       ;         Error Codes: (in AX)
17187                                  <1>       ;            18 - No more files
17188                                  <1>       ;
17189                                  <1>       ; TRDOS 386 (v2.0)
17190                                  <1>       ; 16/10/2016
17191                                  <1>       ;
17192                                  <1>       ; INPUT ->
17193                                  <1>       ;        none
17194                                  <1>       ; OUTPUT ->
17195                                  <1>       ;        EAX = 0 if CH input of 'Find First File' > 0
17196                                  <1>       ;        EAX = First cluster number of file
17197                                  <1>       ;             if CH input of 'Find First File' = 0
17198                                  <1>       ;        EDX = File parameters table/structure address
17199                                  <1>       ;
17200                                  <1>       ;         cf = 1 -> Error code in AL
17201                                  <1>       ;
17202                                  <1>       ; Modified Registers: EAX (at the return of system call)
17203                                  <1>
17204                                  <1>       ;
17205                                  <1>       ; Note: If byte [FFF_Valid] = 0
17206                                  <1>       ;     'sysfnf' will return with 'no more files' error.
17207                                  <1>       ;     If byte [FFF_Valid] = 24
17208                                  <1>       ;     'sysfnf' will return with 32 bytes basic parameters
17209                                  <1>       ;     at the address which is in EDX.
17210                                  <1>       ;     If byte [FFF_Valid] = 128
17211                                  <1>       ;     'sysfnf' will return with 128 bytes Find File
17212                                  <1>       ;     Structure/Table at the address which is in EDX.
17213                                  <1>
```

```
17214 000116A7 803D[5C8D0100]00    <1>       cmp   byte [FFF_Valid], 0
17215 000116AE 7714                <1>       ja    short stsfnf_0
17216                              <1>       ; 'no more files !' error
17217 000116B0 B80C000000          <1>       mov   eax, ERR_NO_MORE_FILES ; 12
17218 000116B5 A3[64030300]        <1>       mov   [u.r0], eax
17219 000116BA A3[C8030300]        <1>       mov   [u.error], eax
17220 000116BF E9ECBCFFFF          <1>       jmp   error
17221                              <1> stsfnf_0:
17222                              <1>       ;cmp  byte [FFF_Valid], 128
17223                              <1>       ;je   short stsfnf_1
17224                              <1>       ;cmp  byte [FFF_Valid], 24
17225                              <1>       ;je   short stsfnf_1
17226                              <1>       ;mov  [FFF_Valid], 24 ; Default
17227                              <1> stsfnf_1:
17228 000116C4 0FB61D[6E810100]    <1>       movzx ebx, byte [Current_Drv]
17229 000116CB 66891D[628D0100]    <1>       mov   [SWP_DRV], bx
17230 000116D2 8A15[0E8A0100]      <1>       mov   dl, [FindFile_Drv]
17231 000116D8 38DA                <1>       cmp   dl, bl
17232 000116DA 750B                <1>       jne   short stsfnf_2
17233 000116DC 86FB                <1>       xchg  bh, bl
17234 000116DE BE00010900          <1>       mov   esi, Logical_DOSDisks
17235 000116E3 01DE                <1>       add   esi, ebx
17236 000116E5 EB0D                <1>       jmp   short sysfnf_3
17237                              <1>
17238                              <1> stsfnf_2:
17239 000116E7 FE05[638D0100]      <1>       inc   byte [SWP_DRV_chg]
17240                              <1>
17241 000116ED E8C064FFFF          <1>       call  change_current_drive
17242 000116F2 7245                <1>       jc    short sysfnf_err_1 ; read error !
17243                              <1>                                ; (do not stop, because
17244                              <1>                                ; we don't have a
17245                              <1>                                ; 'no more files'
17246                              <1>                                ; -file not found- error,
17247                              <1>                                ; next sysfnf system call
17248                              <1>                                ; may solve the problem,
17249                              <1>                                ; after re-placing the disk)
17250                              <1> sysfnf_3:
17251 000116F4 A1[848A0100]        <1>       mov   eax, [FindFile_DirCluster]
17252 000116F9 21C0                <1>       and   eax, eax
17253 000116FB 7550                <1>       jnz   short sysfnf_6
17254                              <1>
17255 000116FD 803D[6D810100]02    <1>       cmp   byte [Current_FATType], 2
17256 00011704 772C                <1>       ja    short sysfnf_err_0 ; invalid, we neeed to stop !?
17257 00011706 803D[6D810100]01    <1>       cmp   byte [Current_FATType], 1
17258 0001170D 7223                <1>       jb    short sysfnf_err_0 ; invalid, we neeed to stop !?
17259                              <1>
17260 0001170F 3805[94880100]      <1>       cmp   byte [DirBuff_ValidData], al ; 0
17261 00011715 7608                <1>       jna   short sysfnf_4
17262                              <1>
17263 00011717 3B05[99880100]      <1>       cmp   eax, [DirBuff_Cluster] ; 0 ?
17264 0001171D 745E                <1>       je    short sysfnf_9
17265                              <1>
17266                              <1>       ;cmp  byte [Current_Dir_Level], 0
17267                              <1>       ;ja   short sysfnf_4
17268                              <1>       ;jna  short sysfnf_9
17269                              <1>
17270                              <1> sysfnf_4:
17271 0001171F FE05[638D0100]      <1>       inc   byte [SWP_DRV_chg]
17272 00011725 E86BB2FFFF          <1>       call  load_FAT_root_directory
17273 0001172A 7351                <1>       jnc   short sysfnf_9
17274                              <1>       ; eax = error code (17, 'drv not ready or read error')
17275 0001172C EB0B                <1>       jmp   short sysfnf_err_1 ; read error ! (no FNF stop)
17276                              <1>                                ; (if you want, try again,
17277                              <1>                                ;  after re-placing the disk)
17278                              <1> sysfnf_5:
17279 0001172E 3C0C                <1>       cmp   al, 12 ; 'no more files' error
17280 00011730 7507                <1>       jne   short sysfnf_err_1 ; (no FNF stop -sysfnf will try
17281                              <1>                                ;  to read the directory again,
17282                              <1>                                ;  if the user calls sysfnf
17283                              <1>                                ;  just after this error return-)
17284                              <1>       ; (FNF stop -sysfnf will not try
17285                              <1>       ;  to read the directory again-)
17286                              <1>
17287                              <1> sysfnf_err_0:
17288 00011732 C605[5C8D0100]00    <1>       mov   byte [FFF_Valid], 0 ; FNF stop sign
17289                              <1> sysfnf_err_1:
17290 00011739 A3[64030300]        <1>       mov   [u.r0], eax
17291 0001173E A3[C8030300]        <1>       mov   [u.error], eax
17292 00011743 E84B130000          <1>       call  reset_working_path
17293 00011748 E963BCFFFF          <1>       jmp   error
17294                              <1>
17295                              <1> sysfnf_6:
17296 0001174D 803D[94880100]00    <1>       cmp   byte [DirBuff_ValidData], 0
17297 00011754 7608                <1>       jna   short sysfnf_7
17298                              <1>
17299 00011756 3B05[99880100]      <1>       cmp   eax, [DirBuff_Cluster]
17300 0001175C 741F                <1>       je    short sysfnf_9
17301                              <1>
17302                              <1> sysfnf_7:
17303 0001175E FE05[638D0100]      <1>       inc   byte [SWP_DRV_chg]
17304 00011764 803D[6D810100]01    <1>       cmp   byte [Current_FATType], 1
17305 0001176B 7309                <1>       jnb   short sysfnf_8
17306                              <1>
17307                              <1>       ; Singlix (TRFS) File System
17308                              <1>       ; (access via compatibility buffer)
17309 0001176D E8EBB2FFFF          <1>       call  load_FS_sub_directory
17310 00011772 7309                <1>       jnc   short sysfnf_9
17311                              <1>
17312 00011774 EBC3                <1>       jmp   short sysfnf_err_1 ; read error (no FNF stop)
17313                              <1>
17314                              <1> sysfnf_8:
17315 00011776 E8A5B2FFFF          <1>       call  load_FAT_sub_directory
17316 0001177B 72BC                <1>       jc    short sysfnf_err_1 ; read error (no FNF stop)
17317                              <1>
17318                              <1> sysfnf_9:
```

```
17319 0001177D E88278FFFF          <1>       call   find_next_file
17320 00011782 72AA                <1>       jc     short sysfnf_5
17321                              <1>
17322 00011784 A0[5D8D0100]        <1>       mov    al, [FFF_Attrib]
17323                              <1>       ;or    al, al
17324                              <1>       ;jz    short sysfnf_10 ; 0 = No filter
17325 00011789 34FF                <1>       xor    al, 0FFh
17326 0001178B 20D8                <1>       and    al, bl
17327 0001178D 75EE                <1>       jnz    short sysfnf_9 ; search for next file until
17328                              <1>                           ; an error return from
17329                              <1>                           ; find_next_file procedure
17330                              <1> sysfnf_10:
17331 0001178F 0FB60D[5C8D0100]    <1>       movzx  ecx, byte [FFF_Valid]
17332 00011796 80F980              <1>       cmp    cl, 128 ; complete FindFile structure/table
17333 00011799 0F84A2FEFFFF        <1>       je     sysfnf_11
17334                              <1>       ;cmp   cl, 24  ; basic parameters
17335                              <1>       ;je    sysfnf_12
17336 0001179F E9AFFEFFFF          <1>       jmp    sysfnf_12
17337                              <1>
17338                              <1> writei:
17339                              <1>       ; 26/10/2016
17340                              <1>       ; 25/10/2016
17341                              <1>       ; 23/10/2016
17342                              <1>       ; 22/10/2016
17343                              <1>       ; 19/10/2016 - TRDOS 386 (TRDOS v2.0)
17344                              <1>       ; 19/05/2015 - 20/05/2015 (Retro UNIX 386 v1)
17345                              <1>       ; 12/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
17346                              <1>       ;
17347                              <1>       ; Write data to file with first cluster number in EAX
17348                              <1>       ;
17349                              <1>       ; INPUTS ->
17350                              <1>       ;    EAX - First cluster number of the file
17351                              <1>       ;    EBX - File number (Open file index number)
17352                              <1>       ;    u.count - byte count to be written
17353                              <1>       ;    u.base - points to user buffer
17354                              <1>       ;    u.fofp - points to dword with current file offset
17355                              <1>       ;    i.size - file size
17356                              <1>       ;    cdev - logical dos drive number of the file
17357                              <1>       ; OUTPUTS ->
17358                              <1>       ;    u.count - cleared
17359                              <1>       ;    u.nread - accumulates total bytes passed back
17360                              <1>       ;    i.size - new file size (if file byte offset overs file size)
17361                              <1>       ;    u.fofp - points to u.off (with new offset value)
17362                              <1>       ;
17363                              <1>       ; (Retro UNIX Prototype : 11/11/2012 - 18/11/2012, UNIXCOPY.ASM)
17364                              <1>       ; ((Modified registers: eax, edx, ebx, ecx, esi, edi, ebp))
17365                              <1>
17366 000117A4 31C9                <1>       xor    ecx, ecx
17367 000117A6 890D[8C030300]      <1>       mov    [u.nread], ecx  ; 0
17368 000117AC 66890D[C4030300]    <1>       mov    [u.pcount], cx ; 19/05/2015
17369 000117B3 390D[88030300]      <1>       cmp    [u.count], ecx
17370 000117B9 7701                <1>       ja     short writei_1
17371 000117BB C3                  <1>       retn
17372                              <1> writei_1:
17373 000117BC 881D[1C8D0100]      <1>       mov    [writei.ofn], bl ; Open file number
17374 000117C2 880D[578D0100]      <1>       mov    [setfmod], cl ; 0 ; reset 'update lm date&time' sign
17375                              <1> dskw_0:
17376                              <1>       ; 26/10/2016
17377                              <1>       ; 22/10/2016, 23/10/2016, 25/10/2016
17378                              <1>       ; 19/10/2016 - TRDOS 386 (TRDOS v2.0)
17379                              <1>       ; 31/05/2015 - 25/07/2015 (Retro UNIX 386 v1)
17380                              <1>       ; 26/04/2013 - 20/09/2013 (Retro UNIX 8086 v1)
17381                              <1>       ;
17382                              <1>       ; 01/08/2013 (mkdir_w check)
17383 000117C8 E8D7000000         <1>       call   mget_w
17384                              <1>       ; eax = sector/block number
17385                              <1>
17386 000117CD 8B1D[74030300]      <1>       mov    ebx, [u.fofp]
17387 000117D3 8B13                <1>       mov    edx, [ebx]
17388 000117D5 81E2FF010000        <1>       and    edx, 1FFh  ; / test the lower 9 bits of the file offset
17389 000117DB 750C                <1>       jnz    short dskw_1 ; / if its non-zero, branch
17390                              <1>                           ; if zero, file offset = 0,
17391                              <1>                                ; / 512, 1024,...(i.e., start of new block)
17392 000117DD 813D[88030300]0002- <1>       cmp    dword [u.count], 512
17392 000117E5 0000                <1>
17393                              <1>                           ; / if zero, is there enough data to fill
17394                              <1>                           ; / an entire block? (i.e., no. of
17395 000117E7 7337                <1>       jnb    short dskw_2 ; / bytes to be written greater than 512.?
17396                              <1>                          ; / Yes, branch. Don't have to read block
17397                              <1> dskw_1: ; in as no past info. is to be saved
17398                              <1>       ; (the entire block will be overwritten).
17399                              <1>       ; 23/10/2016
17400                              <1>
17401 000117E9 BB[7C070300]        <1>       mov    ebx, writei_buffer
17402                              <1>       ; esi = logical dos drive description table address
17403                              <1>       ; eax = sector number
17404                              <1>       ; ebx = buffer address (in kernel's memory space)
17405                              <1>       ; ecx = sector count
17406 000117EE B901000000          <1>       mov    ecx, 1
17407 000117F3 E8A30D0000          <1>       call   disk_read
17408                              <1>       ;call  dskrd  ; / no, must retain old info..
17409                              <1>                           ; / Hence, read block 'r1' into an I/O buffer
17410 000117F8 7326                <1>       jnc    short dskw_2
17411                              <1>
17412                              <1>       ; disk read error
17413 000117FA B811000000          <1>       mov    eax, 17 ; drive not ready or READ ERROR !
17414                              <1> dskw_err: ; jump from disk write error
17415 000117FF A3[64030300]        <1>       mov    [u.r0], eax
17416 00011804 A3[C8030300]        <1>       mov    [u.error], eax
17417                              <1>
17418 00011809 803D[578D0100]00    <1>       cmp    byte [setfmod], 0
17419 00011810 0F869ABBFFFF        <1>       jna    error
17420                              <1>
17421 00011816 E8AF030000          <1>       call   update_file_lmdt ; update last modif. date&time of the file
17422                              <1>       ;mov   byte [setfmod], 0
```

```
17423                                  <1>
17424 0001181B E990BBFFFF              <1>       jmp    error
17425                                  <1>
17426                                  <1> dskw_2: ; 3:
17427                                  <1>       ; 23/10/2016
17428 00011820 C605[F88C0100]01        <1>       mov    byte [writei.valid], 1 ; writei buffer contains valid data
17429 00011827 56                      <1>       push   esi ; logical dos drive description table address
17430                                  <1>       ; EAX (r1) = block/sector number
17431                                  <1>       ;call  wslot
17432                                  <1>              ; jsr r0,wslot / set write and inhibit bits in I/O queue,
17433                                  <1>                       ; / proc. status=0, r5 points to 1st word of data
17434 00011828 803D[C6030300]00        <1>       cmp    byte [u.kcall], 0
17435 0001182F 770F                    <1>       ja     short dskw_4 ; zf=0 -> the caller is 'mkdir'
17436                                  <1>       ;
17437 00011831 66833D[C4030300]00      <1>       cmp    word [u.pcount], 0
17438 00011839 7705                    <1>       ja     short dskw_4
17439                                  <1> dskw_3:
17440                                  <1>       ; [u.base] = virtual address to transfer (as source address)
17441 0001183B E821FAFFFF              <1>       call   trans_addr_r ; translate virtual address to physical (r)
17442                                  <1> dskw_4:
17443 00011840 BB[7C070300]            <1>       mov    ebx, writei_buffer
17444                                  <1>       ; EBX (r5) = system (I/O) buffer address
17445 00011845 E883FAFFFF              <1>       call   sioreg
17446                                  <1>       ; ESI = file (user data) offset
17447                                  <1>       ; EDI = sector (I/O) buffer offset
17448                                  <1>       ; ECX = byte count
17449                                  <1>       ;
17450 0001184A F3A4                    <1>       rep    movsb
17451                                  <1>       ; 25/07/2015
17452                                  <1>       ; eax = remain bytes in buffer
17453                                  <1>       ;       (check if remain bytes in the buffer > [u.pcount])
17454 0001184C 09C0                    <1>       or     eax, eax
17455 0001184E 75EB                    <1>       jnz    short dskw_3 ; (page end before system buffer end!)
17456                                  <1>
17457                                  <1>       ; 23/10/2016
17458 00011850 B101                    <1>       mov    cl, 1
17459 00011852 5E                      <1>       pop    esi
17460 00011853 A1[FC8C0100]            <1>       mov    eax, [writei.sector]
17461                                  <1>       ; esi = logical dos drive description table address
17462                                  <1>       ; eax = sector number
17463                                  <1>       ; ebx = writei buffer address
17464                                  <1>       ; ecx = sector count
17465 00011858 E82F0D0000              <1>       call   disk_write ; / yes, write the block
17466 0001185D 7307                    <1>       jnc    short dskw_5
17467                                  <1>
17468 0001185F B812000000              <1>       mov    eax, 18 ; drive not ready or WRITE ERROR !
17469 00011864 EB99                    <1>       jmp    short dskw_err
17470                                  <1>
17471                                  <1> dskw_5:
17472                                  <1>       ; 26/10/2016
17473 00011866 0FB61D[1C8D0100]        <1>       movzx  ebx, byte [writei.ofn] ; open file number
17474 0001186D C0E302                  <1>       shl    bl, 2 ; *4
17475 00011870 8B83[B88D0100]          <1>       mov    eax, [ebx+OF_POINTER]
17476 00011876 3B83[E08D0100]          <1>       cmp    eax, [ebx+OF_SIZE]
17477 0001187C 7606                    <1>       jna    short dskw_6
17478 0001187E 8983[E08D0100]          <1>       mov    [ebx+OF_SIZE], eax
17479                                  <1> dskw_6:
17480                                  <1>       ;shr   bl, 2
17481 00011884 833D[88030300]00        <1>       cmp    dword [u.count], 0 ; / any more data to write?
17482 0001188B 760A                    <1>       jna    short dskw_7
17483 0001188D A1[0C8D0100]            <1>       mov    eax, [writei.fclust]
17484 00011892 E931FFFFFF              <1>       jmp    dskw_0 ; / yes, branch
17485                                  <1> dskw_7:
17486                                  <1>       ; update last modif. date&time of the file
17487                                  <1>       ; (also updates file size as OF_SIZE)
17488 00011897 E82E030000              <1>       call   update_file_lmdt
17489                                  <1>       ;mov   byte [setfmod], 0
17490                                  <1>
17491                                  <1>       ; 03/08/2013
17492 0001189C C605[C6030300]00        <1>       mov    byte [u.kcall], 0
17493                                  <1>       ; 23/10/2016
17494                                  <1>       ;mov   eax, [writei.fclust]
17495 000118A3 C3                      <1>       retn
17496                                  <1>
17497                                  <1> mget_w:
17498                                  <1>       ; 02/11/2016
17499                                  <1>       ; 01/11/2016
17500                                  <1>       ; 23/10/2016, 31/10/2016
17501                                  <1>       ; 22/10/2016 - TRDOS 386 (TRDOS v2.0)
17502                                  <1>       ; 03/06/2015 (Retro UNIX 386 v1, 'mget', u.5s)
17503                                  <1>       ; 22/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
17504                                  <1>       ;
17505                                  <1>       ; Get existing or (allocate) a new disk block for file
17506                                  <1>       ;
17507                                  <1>       ; INPUTS ->
17508                                  <1>       ;    [u.fofp] = file offset pointer
17509                                  <1>       ;    [i.size] = file size
17510                                  <1>       ;    [u.count] = byte count
17511                                  <1>       ;    EAX = First cluster
17512                                  <1>       ;    [cdev] = Logical dos drive number
17513                                  <1>       ;    [writei.ofn] = File Number
17514                                  <1>       ;                (Open file index, 0 based)
17515                                  <1>       ;    ([u.off] = file offset)
17516                                  <1>       ; OUTPUTS ->
17517                                  <1>       ;    EAX = logical sector number
17518                                  <1>       ;    ESI = Logical Dos Drive Description Table address
17519                                  <1>       ;
17520                                  <1>       ; Modified registers: EDX, EBX, ECX, ESI, EDI, EBP
17521                                  <1>
17522 000118A4 8B35[74030300]          <1>       mov    esi, [u.fofp]
17523 000118AA 8B2E                    <1>       mov    ebp, [esi] ; u.off (or EBX*4+OF_POINTER)
17524                                  <1>
17525 000118AC 29C9                    <1>       sub    ecx, ecx
17526 000118AE 8A2D[46030300]          <1>       mov    ch, [cdev]
17527                                  <1>
```

```
17528 000118B4 BE00010900          <1>       mov   esi, Logical_DOSDisks
17529 000118B9 01CE                <1>       add   esi, ecx
17530                              <1>
17531                              <1>       ; 31/10/2016
17532 000118BB 89C3                <1>       mov   ebx, eax ; First Cluster or FDT address
17533                              <1>
17534 000118BD 807E0300            <1>       cmp   byte [esi+LD_FATType], 0
17535 000118C1 0F86DD010000        <1>       jna   mget_w_14 ; Singlix FS
17536                              <1>
17537 000118C7 0FB74611            <1>       movzx  eax, word [esi+LD_BPB+BytesPerSec]
17538 000118CB 0FB65613            <1>       movzx  edx, byte [esi+LD_BPB+SecPerClust]
17539 000118CF 8815[FA8C0100]      <1>       mov   [writei.spc], dl  ; sectors per cluster
17540 000118D5 F7E2                <1>       mul   edx
17541                              <1>       ; edx = 0
17542                              <1>       ; eax = bytes per cluster (<= 65536)
17543                              <1>
17544                              <1>       ; 02/11/2016
17545 000118D7 89C1                <1>       mov   ecx, eax
17546 000118D9 48                  <1>       dec   eax
17547 000118DA 66A3[008D0100]      <1>       mov   [writei.bpc], ax
17548                              <1>
17549 000118E0 89E8                <1>       mov   eax, ebp
17550 000118E2 0305[88030300]      <1>       add   eax, [u.count] ; next file position
17551 000118E8 3B05[55040300]      <1>       cmp   eax, [i.size] ; <= file size ?
17552 000118EE 0F86FC000000        <1>       jna   mget_w_4 ; no
17553                              <1>
17554 000118F4 F7F1                <1>       div   ecx
17555 000118F6 A3[088D0100]        <1>       mov   [writei.c_index], eax ; cluster index
17556                              <1>       ; edx = byte offset in cluster (<= 65535)
17557                              <1>       ;mov  [writei.offset], dx
17558                              <1>       ;shr  dx, 9 ; / 512
17559                              <1>       ;mov  [writei.s_index], dl ; sector index in cluster (0 to spc -1)
17560                              <1>
17561 000118FB 29D2               <1>       sub   edx, edx ; 01/11/2016
17562 000118FD 8915[FC8C0100]      <1>       mov   [writei.sector], edx ; 0
17563 00011903 668915[028D0100]    <1>       mov   [writei.offset], dx  ; byte offset in cluster
17564 0001190A 8815[FB8C0100]      <1>       mov   [writei.s_index], dl ; sector index in cluster (0 to spc -1)
17565                              <1>
17566 00011910 89D8                <1>       mov   eax, ebx ; First Cluster
17567                              <1>
17568                              <1>       ; is this the 1st mget_w or a next mget_w call ? (by 'writei')
17569 00011912 3815[F88C0100]      <1>       cmp   byte [writei.valid], dl ; 0
17570 00011918 7624                <1>       jna   short mget_w_0
17571                              <1>
17572 0001191A 8815[F88C0100]      <1>       mov   byte [writei.valid], dl ; 0 ; reset ('writei' will set it)
17573                              <1>
17574 00011920 3B05[0C8D0100]      <1>       cmp   eax, [writei.fclust]
17575 00011926 7516                <1>       jne   short mget_w_0
17576                              <1>
17577 00011928 8A0D[46030300]      <1>       mov   cl, [cdev]
17578 0001192E 3A0D[F98C0100]      <1>       cmp   cl, [writei.drv]
17579 00011934 7508                <1>       jne   short mget_w_0
17580                              <1>       ; [writei.l_clust] & [writei.l_index] are valid,
17581                              <1>       ;  we don't need to get last cluster & last cluster index
17582 00011936 8B0D[188D0100]      <1>       mov   ecx, [writei.l_index]
17583 0001193C EB64                <1>       jmp   short mget_w_2
17584                              <1> mget_w_0:
17585 0001193E A3[0C8D0100]        <1>       mov   [writei.fclust], eax ; first cluster
17586                              <1>       ; edx = 0
17587 00011943 A3[048D0100]        <1>       mov   [writei.cluster], eax ; first cluster ; 01/11/2016
17588 00011948 8915[108D0100]      <1>       mov   [writei.fs_index], edx ; 0 ; curret cluster index
17589                              <1>
17590                              <1>       ; FAT file system (FAT12, FAT16, FAT32)
17591 0001194E E8E2B6FFFF          <1>       call  get_last_cluster
17592 00011953 0F822B010000        <1>       jc    mget_w_err ; eax = error code
17593                              <1>
17594 00011959 A3[148D0100]        <1>       mov   [writei.lclust], eax ; last cluster
17595                              <1>
17596 0001195E 8B0D[388B0100]      <1>       mov   ecx, [glc_index] ; last cluster index
17597 00011964 890D[188D0100]      <1>       mov   [writei.l_index], ecx
17598                              <1>
17599 0001196A A0[1C8D0100]        <1>       mov   al, [writei.ofn]
17600 0001196F FEC0                <1>       inc   al
17601 00011971 A2[578D0100]        <1>       mov   [setfmod], al ; update lm date&time sign
17602                              <1>
17603                              <1> mget_w_1:
17604 00011976 3B0D[088D0100]      <1>       cmp   ecx, [writei.c_index]  ; last cluster index
17605 0001197C 7324                <1>       jnb   short mget_w_2 ; 01/11/2016
17606                              <1>
17607 0001197E A1[148D0100]        <1>       mov   eax, [writei.lclust]
17608                              <1>       ; EAX = Last cluster
17609 00011983 E8BBB7FFFF          <1>       call  add_new_cluster
17610 00011988 0F82F6000000        <1>       jc    mget_w_err ; eax = error code
17611                              <1>       ; edx = 0
17612 0001198E A3[148D0100]        <1>       mov   [writei.lclust], eax ; (new) last cluster
17613 00011993 8B0D[188D0100]      <1>       mov   ecx, [writei.l_index]
17614 00011999 41                  <1>       inc   ecx ; add 1 to last cluster index
17615 0001199A 890D[188D0100]      <1>       mov   [writei.l_index], ecx ; current last cluster index
17616                              <1>
17617 000119A0 EBD4                <1>       jmp   short mget_w_1
17618                              <1>
17619                              <1> mget_w_2:
17620 000119A2 89E9                <1>       mov   ecx, ebp
17621 000119A4 030D[88030300]      <1>       add   ecx, [u.count]
17622 000119AA 890D[55040300]      <1>       mov   [i.size], ecx ; save new file size
17623                              <1>       ;sub  edx, edx ; 0
17624                              <1>
17625 000119B0 A0[46030300]        <1>       mov   al, [cdev]
17626 000119B5 A2[F98C0100]        <1>       mov   [writei.drv], al ; physical drive number
17627                              <1>       ; edx = 0
17628 000119BA 89E8                <1>       mov   eax, ebp ; file offset
17629 000119BC 0FB70D[008D0100]    <1>       movzx  ecx, word [writei.bpc] ; bytes per cluster - 1
17630 000119C3 41                  <1>       inc   ecx ; bytes per cluster
17631 000119C4 F7F1                <1>       div   ecx
17632                              <1>       ; edx = byte offset in cluster (<= 65535)
```

```
17633                                  <1>       ; eax = cluster index
17634 000119C6 A3[088D0100]            <1>       mov    [writei.c_index], eax
17635 000119CB 668915[028D0100]        <1>       mov    [writei.offset], dx
17636 000119D2 66C1EA09                <1>       shr    dx, 9 ; / 512
17637 000119D6 8815[FB8C0100]          <1>       mov    [writei.s_index], dl ; sector index in cluster (0 to spc -1)
17638                                  <1>
17639                                  <1> mget_w_3:
17640 000119DC 3B05[188D0100]          <1>       cmp    eax, [writei.l_index] ; last cluster index
17641 000119E2 752A                    <1>       jne    short mget_w_5
17642                                  <1>
17643 000119E4 A3[108D0100]            <1>       mov    [writei.fs_index], eax ; cluster index (for next check)
17644 000119E9 A1[148D0100]            <1>       mov    eax, [writei.lclust] ; last cluster
17645 000119EE EB60                    <1>       jmp    short mget_w_10
17646                                  <1>
17647                                  <1> mget_w_4: ; 02/11/2016
17648                                  <1>       ; eax = next file position
17649 000119F0 2B05[88030300]          <1>       sub    eax, [u.count] ; current file position
17650                                  <1>       ; edx = 0
17651                                  <1>       ; ecx = bytes per cluster
17652 000119F6 F7F1                    <1>       div    ecx
17653 000119F8 A3[088D0100]            <1>       mov    [writei.c_index], eax ; cluster index
17654 000119FD 668915[028D0100]        <1>       mov    [writei.offset], dx
17655 00011A04 66C1EA09                <1>       shr    dx, 9 ; / 512
17656 00011A08 8815[FB8C0100]          <1>       mov    [writei.s_index], dl ; sector index in cluster (0 to spc -1)
17657                                  <1>
17658                                  <1> mget_w_5:
17659 00011A0E 21C0                    <1>       and    eax, eax ; 0 = First Cluster's index number
17660 00011A10 750C                    <1>       jnz    short mget_w_6
17661                                  <1>
17662 00011A12 A3[108D0100]            <1>       mov    [writei.fs_index], eax ; cluster index (for next check)
17663 00011A17 A1[0C8D0100]            <1>       mov    eax, [writei.fclust] ; first cluster
17664 00011A1C EB32                    <1>       jmp    short mget_w_10
17665                                  <1>
17666                                  <1> mget_w_6:
17667 00011A1E 3B05[108D0100]          <1>       cmp    eax, [writei.fs_index] ; current cluster index (>0)
17668 00011A24 7507                    <1>       jne    short mget_w_7
17669 00011A26 A1[048D0100]            <1>       mov    eax, [writei.cluster] ; current cluster
17670 00011A2B EB3A                    <1>       jmp    short mget_w_11
17671                                  <1>
17672                                  <1> mget_w_7:
17673 00011A2D 89C1                    <1>       mov    ecx, eax
17674 00011A2F 2B0D[108D0100]          <1>       sub    ecx, [writei.fs_index]
17675 00011A35 730D                    <1>       jnc    short mget_w_8
17676                                  <1>       ; get cluster by index from the first cluster
17677 00011A37 A1[0C8D0100]            <1>       mov    eax, [writei.fclust]
17678 00011A3C 8B0D[088D0100]          <1>       mov    ecx, [writei.c_index]
17679 00011A42 EB05                    <1>       jmp    short mget_w_9
17680                                  <1>
17681                                  <1> mget_w_8:
17682 00011A44 A1[048D0100]            <1>       mov    eax, [writei.cluster] ; beginning cluster
17683                                  <1>       ; ecx = cluster sequence number after the beginning cluster
17684                                  <1>       ; sub  edx, edx ; 0
17685                                  <1>
17686                                  <1> mget_w_9:
17687                                  <1>       ; EAX = Beginning cluster
17688                                  <1>       ; EDX = Sector index in disk/file section
17689                                  <1>       ;       (Only for SINGLIX file system!)
17690                                  <1>       ; ECX = Cluster sequence number after the beginning cluster
17691                                  <1>       ; ESI = Logical DOS Drive Description Table address
17692 00011A49 E8FBB7FFFF              <1>       call   get_cluster_by_index
17693 00011A4E 7234                    <1>       jc     short mget_w_err ; error code in EAX
17694                                  <1>       ; EAX = Cluster number
17695                                  <1> mget_w_10:
17696 00011A50 A3[048D0100]            <1>       mov    [writei.cluster], eax ; FDT number for Singlix File System
17697                                  <1>
17698 00011A55 807E0300                <1>       cmp    byte [esi+LD_FATType], 0
17699 00011A59 7638                    <1>       jna    short mget_w_13
17700                                  <1>       ; 01/11/2016
17701 00011A5B 8B15[088D0100]          <1>       mov    edx, [writei.c_index]
17702 00011A61 8915[108D0100]          <1>       mov    [writei.fs_index], edx
17703                                  <1> mget_w_11:
17704 00011A67 83E802                  <1>       sub    eax, 2
17705 00011A6A 0FB615[FA8C0100]        <1>       movzx  edx, byte [writei.spc]
17706 00011A71 F7E2                    <1>       mul    edx
17707                                  <1>
17708 00011A73 034668                  <1>       add    eax, [esi+LD_DATABegin]
17709 00011A76 8A15[FB8C0100]          <1>       mov    dl, [writei.s_index]
17710 00011A7C 01D0                    <1>       add    eax, edx
17711                                  <1> mget_w_12:
17712 00011A7E A3[FC8C0100]            <1>       mov    [writei.sector], eax
17713                                  <1>       ;; buffer validation must be done in writei
17714                                  <1>       ;;mov  byte [writei.valid], 1
17715 00011A83 C3                      <1>       retn
17716                                  <1>
17717                                  <1> mget_w_err:
17718 00011A84 A3[C8030300]            <1>       mov    [u.error], eax
17719 00011A89 A3[64030300]            <1>       mov    [u.r0], eax
17720 00011A8E E91DB9FFFF              <1>       jmp    error
17721                                  <1>
17722                                  <1> mget_w_13:
17723                                  <1>       ; EAX = FDT number (Current Section)
17724                                  <1>       ; EDX = Sector index from the first section (0,1,2,3,4...)
17725 00011A93 2B15[108D0100]          <1>       sub    edx, [writei.fs_index]
17726                                  <1>       ; EDX = Sector index from current section
17727 00011A99 8915[108D0100]          <1>       mov    [writei.fs_index], edx
17728 00011A9F 40                      <1>       inc    eax ; the first data sector in FS disk section
17729 00011AA0 01D0                    <1>       add    eax, edx
17730 00011AA2 EBDA                    <1>       jmp    short mget_w_12
17731                                  <1>
17732                                  <1> mget_w_14:
17733 00011AA4 8A4E12                  <1>       mov    cl, [esi+LD_FS_BytesPerSec+1]
17734 00011AA7 D0E9                    <1>       shr    cl, 1 ;  ; 1 for 512 bytes, 4 for 2048 bytes
17735 00011AA9 880D[FA8C0100]          <1>       mov    [writei.spc], cl  ; sectors per cluster
17736                                  <1>       ; NOTE: writei bytes per sector value is always 512 !
17737 00011AAF 66C705[008D0100]00-     <1>       mov    word [writei.bpc], 512
```

```
17737 00011AB7 02                <1>
17738                            <1>
17739 00011AB8 89E9              <1>       mov   ecx, ebp
17740 00011ABA 030D[88030300]    <1>       add   ecx, [u.count] ; next file position
17741 00011AC0 3B0D[55040300]    <1>       cmp   ecx, [i.size] ; <= file size ?
17742 00011AC6 0F86C8000000      <1>       jna   mget_w_19 ; no
17743                            <1>
17744 00011ACC 29D2              <1>       sub   edx, edx ; 0
17745 00011ACE 8915[FC8C0100]    <1>       mov   [writei.sector], edx ; 0
17746 00011AD4 668915[028D0100]  <1>       mov   [writei.offset], dx  ; byte offset in cluster
17747 00011ADB 8815[FB8C0100]    <1>       mov   [writei.s_index], dl ; sector index in cluster (0 to spc -1)
17748                            <1>
17749 00011AE1 C1E909            <1>       shr   ecx, 9 ; 1 cluster = 512 bytes
17750 00011AE4 890D[088D0100]    <1>       mov   [writei.c_index], ecx ; section/cluster index
17751                            <1>
17752 00011AEA 89D8              <1>       mov   eax, ebx ; FDT number (First FDT address)
17753                            <1>
17754                            <1>       ; is this the 1st mget_w or a next mget_w call ? (by 'writei')
17755 00011AEC 3815[F88C0100]    <1>       cmp   byte [writei.valid], dl ; 0
17756 00011AF2 7624              <1>       jna   short mget_w_15
17757                            <1>
17758 00011AF4 8815[F88C0100]    <1>       mov   byte [writei.valid], dl ; 0 ; reset ('writei' will set it)
17759                            <1>
17760 00011AFA 3B05[0C8D0100]    <1>       cmp   eax, [writei.fclust]
17761 00011B00 7516              <1>       jne   short mget_w_15
17762                            <1>
17763 00011B02 8A0D[46030300]    <1>       mov   cl, [cdev]
17764 00011B08 3A0D[F98C0100]    <1>       cmp   cl, [writei.drv]
17765 00011B0E 7508              <1>       jne   short mget_w_15
17766                            <1>       ; [writei.l_clust] & [writei.l_index] are valid,
17767                            <1>       ;  we don't need to get last cluster & last cluster index
17768 00011B10 8B0D[188D0100]    <1>       mov   ecx, [writei.l_index]
17769 00011B16 EB49              <1>       jmp   short mget_w_17
17770                            <1> mget_w_15:
17771 00011B18 A3[0C8D0100]      <1>       mov   [writei.fclust], eax ; first section (FDT number)
17772                            <1>       ; edx = 0
17773 00011B1D 8915[048D0100]    <1>       mov   [writei.cluster], edx ; 0 ; current section
17774 00011B23 8915[108D0100]    <1>       mov   [writei.fs_index], edx ; 0 ; curret section index
17775                            <1>
17776                            <1>       ; eax = FDT number (section 0 header address)
17777 00011B29 E845B7FFFF        <1>       call  get_last_section
17778 00011B2E 0F8250FFFFFF      <1>       jc    mget_w_err ; eax = error code
17779                            <1>
17780 00011B34 8915[108D0100]    <1>       mov   [writei.fs_index], edx ; sector index in last section
17781                            <1>
17782 00011B3A A3[148D0100]      <1>       mov   [writei.lclust], eax ; last section address
17783                            <1>
17784 00011B3F 8B0D[388B0100]    <1>       mov   ecx, [glc_index] ; last section index
17785 00011B45 890D[188D0100]    <1>       mov   [writei.l_index], ecx
17786                            <1>
17787 00011B4B A0[1C8D0100]      <1>       mov   al, [writei.ofn]
17788 00011B50 FEC0              <1>       inc   al
17789 00011B52 A2[578D0100]      <1>       mov   [setfmod], al ; update lm date&time sign
17790                            <1>
17791                            <1> mget_w_16:
17792                            <1>       ; edx = (existing) last section (sector) index
17793 00011B57 8B0D[088D0100]    <1>       mov   ecx, [writei.c_index] ; final section (sector) index
17794 00011B5D 29D1              <1>       sub   ecx, edx
17795 00011B5F 7633              <1>       jna   short mget_w_19
17796                            <1>        ; ecx = sector count
17797                            <1> mget_w_17:
17798 00011B61 A1[148D0100]      <1>       mov   eax, [writei.lclust]
17799                            <1>       ; ESI = Logical dos drv desc. table address
17800                            <1>       ; EAX = Last section
17801                            <1>       ; (ECX = 0 for directory)
17802                            <1>       ; ECX = sector count (except FDT)
17803 00011B66 E8CBACFFFF        <1>       call  add_new_fs_section
17804 00011B6B 7312              <1>       jnc   short mget_w_18
17805                            <1>
17806                            <1>       ; If error number = 27h (insufficient disk space)
17807                            <1>       ; it is needed to check free consequent sectors
17808                            <1>       ; (1 data sector at least and +1 section header sector)
17809                            <1>
17810 00011B6D 83F827            <1>       cmp   eax, 27h
17811 00011B70 0F850EFFFFFF      <1>       jne   mget_w_err ; eax = error code
17812                            <1>
17813                            <1>       ; ecx = count of free consequent sectors
17814                            <1>       ; ecx must be > 1 (1 data + 1 header sector)
17815 00011B76 49                <1>       dec   ecx
17816 00011B77 0F8407FFFFFF      <1>       jz    mget_w_err
17817 00011B7D EBE2              <1>       jmp   short mget_w_17
17818                            <1>
17819                            <1> mget_w_18:
17820 00011B7F A3[148D0100]      <1>       mov   [writei.lclust], eax ; (new) last section
17821                            <1>       ; ecx = sector count (except section header)
17822 00011B84 8B15[188D0100]    <1>       mov   edx, [writei.l_index]
17823 00011B8A 01CA              <1>       add   edx, ecx ; add sector count to index
17824 00011B8C 8915[188D0100]    <1>       mov   [writei.l_index], edx
17825 00011B92 EBC3              <1>       jmp   short mget_w_16
17826                            <1>
17827                            <1> mget_w_19:
17828 00011B94 89E9              <1>       mov   ecx, ebp
17829 00011B96 030D[88030300]    <1>       add   ecx, [u.count]
17830 00011B9C 890D[55040300]    <1>       mov   [i.size], ecx ; save new file size
17831                            <1>       ;sub   edx, edx ; 0
17832                            <1>
17833 00011BA2 A0[46030300]      <1>       mov   al, [cdev]
17834 00011BA7 A2[F98C0100]      <1>       mov   [writei.drv], al ; physical drive number
17835                            <1>       ; edx = 0
17836 00011BAC 89E8              <1>       mov   eax, ebp ; file offset
17837 00011BAE 89C2              <1>       mov   edx, eax
17838                            <1>       ; 1 cluster = 512 bytes (for Singlix FS)
17839 00011BB0 C1E809            <1>       shr   eax, 9  ; / 512
17840 00011BB3 81E2FF010000      <1>       and   edx, 1FFh
17841                            <1>       ; edx = byte offset in cluster/sector (<= 511)
```

```
                                <1>       ; eax = section (sector/cluster) index
00011BB9 A3[088D0100]           <1>       mov    [writei.c_index], eax
00011BBE 668915[028D0100]       <1>       mov    [writei.offset], dx
                                <1>       ;mov   byte [writei.s_index], 0 ; sector index in cluster
00011BC5 E912FEFFFF             <1>       jmp    mget_w_3
                                <1>
                                <1> update_file_lmdt: ; & update file size
                                <1>       ; 26/10/2016
                                <1>       ; 24/10/2016
                                <1>       ; 23/10/2016
                                <1>       ; 22/10/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ;
                                <1>       ; Update last modification date&time of file
                                <1>       ; (call from syswrite -> writei)
                                <1>       ; ((also updates file size)) // 26/10/2016
                                <1>       ;
                                <1>       ; INPUT:
                                <1>       ;      byte [setfmod] = open file number
                                <1>       ; OUTPUT:
                                <1>       ;      cf = 0 -> success !
                                <1>       ;      cf = 1 -> lmdt update has been failed!
                                <1>       ;
                                <1>       ; Modified registers: eax, ebx, ecx, edx, esi, edi
                                <1>       ;
                                <1>
                                <1>       ;cmp   byte [setfmod], 0
                                <1>       ;jna   short uflmdt_2 ; nothing to do
                                <1>
00011BCA 31C0                   <1>       xor    eax, eax
                                <1>
00011BCC 0FB61D[578D0100]       <1>       movzx  ebx, byte [setfmod]
00011BD3 FECB                   <1>       dec    bl ; open file index number (0 based)
                                <1>
00011BD5 8AA3[908D0100]         <1>       mov    ah, [ebx+OF_DRIVE]
00011BDB BE00010900             <1>       mov    esi, Logical_DOSDisks
00011BE0 01C6                   <1>       add    esi, eax
00011BE2 C0E302                 <1>       shl    bl, 2 ; *4
00011BE5 8B8B[688D0100]         <1>       mov    ecx, [ebx+OF_FCLUSTER] ; first cluster
00011BEB 8B93[308E0100]         <1>       mov    edx, [ebx+OF_DIRCLUSTER] ; dir cluster
                                <1>
00011BF1 D0EB                   <1>       shr    bl, 1 ; /2
00011BF3 0FB7BB[D08E0100]       <1>       movzx  edi, word [ebx+OF_DIRENTRY]
                                <1>
00011BFA 803D[94880100]01       <1>       cmp    byte [DirBuff_ValidData], 1
00011C01 726E                   <1>       jb     short uflmdt_4
                                <1>
00011C03 A0[92880100]           <1>       mov    al, [DirBuff_DRV]
00011C08 2C41                   <1>       sub    al, 'A'
00011C0A 38E0                   <1>       cmp    al, ah
00011C0C 7563                   <1>       jne    short uflmdt_4 ; different drive
00011C0E 8A4603                 <1>       mov    al, [esi+LD_FATType]
00011C11 3A05[93880100]         <1>       cmp    al, [DirBuff_FATType]
00011C17 755B                   <1>       jne    short uflmdt_5 ; different FS type
00011C19 3B15[99880100]         <1>       cmp    edx, [DirBuff_Cluster]
00011C1F 7553                   <1>       jne    short uflmdt_5 ; different cluster
                                <1>
                                <1> uflmdt_1:
                                <1>       ; Directory buffer is ready here!
                                <1>       ; OF_FCLUSTER must be compared/verified
00011C21 BE00000800             <1>       mov    esi, Directory_Buffer
00011C26 66C1E705               <1>       shl    di, 5 ; dir entry index * 32
00011C2A 01FE                   <1>       add    esi, edi ; offset
                                <1>       ;
00011C2C F6460B18               <1>       test   byte [esi+DirEntry_Attr], 18h ; Vol & Dir
00011C30 750F                   <1>       jnz    short uflmdt_2 ; not a valid file !
00011C32 668B4614               <1>       mov    ax, [esi+DirEntry_FstClusHI]
00011C36 C1E010                 <1>       shl    eax, 16
00011C39 668B461A               <1>       mov    ax, [esi+DirEntry_FstClusLO]
00011C3D 39C8                   <1>       cmp    eax, ecx ; same first cluster ?
00011C3F 7407                   <1>       je     short uflmdt_3 ; yes, it is OK !!!
                                <1>
                                <1> uflmdt_2:
                                <1>       ; save directory buffer if has modified/changed sign
                                <1>       ; (It is good to save dir buff even if the searched
                                <1>       ; directory entry is not found !?)
00011C41 E84499FFFF             <1>       call   save_directory_buffer
00011C46 F9                     <1>       stc    ; update failed
00011C47 C3                     <1>       retn
                                <1>
                                <1> uflmdt_3:
                                <1>       ; Update directory entry
                                <1>       ; 26/10/2016
00011C48 D0E3                   <1>       shl    bl, 1 ; *2
00011C4A 8B83[E08D0100]         <1>       mov    eax, [ebx+OF_SIZE] ; file size
00011C50 89461C                 <1>       mov    [esi+DirEntry_FileSize], eax
                                <1>       ;
00011C53 E89498FFFF             <1>       call   convert_current_date_time
                                <1>       ; OUTPUT -> DX = Date in dos dir entry format
                                <1>       ;           AX = Time in dos dir entry format
00011C58 66894616               <1>       mov    [esi+DirEntry_WrtTime], ax
00011C5C 66895618               <1>       mov    [esi+DirEntry_WrtDate], dx
00011C60 66895612               <1>       mov    [esi+DirEntry_LastAccDate], dx
00011C64 C605[94880100]02       <1>       mov    byte [DirBuff_ValidData], 2
00011C6B E81A99FFFF             <1>       call   save_directory_buffer
00011C70 C3                     <1>       retn
                                <1>
                                <1> uflmdt_4:
                                <1>       ; Directory buffer sector read&write
                                <1>       ; 23/10/2016
                                <1>       ;
00011C71 8A4603                 <1>       mov    al, [esi+LD_FATType]
                                <1> uflmdt_5:
00011C74 BB[84090300]           <1>       mov    ebx, rw_buffer ; Common r/w sector buffer addr
                                <1>
00011C79 20C0                   <1>       and    al, al ; 0 = Singlix FS
```

```
17947 00011C7B 0F8492000000        <1>       jz    uflmdt_11
17948                              <1>
17949 00011C81 21D2                <1>       and   edx, edx
17950 00011C83 7521                <1>       jnz   short uflmdt_9
17951                              <1>
17952 00011C85 3C02                <1>       cmp   al, 2  ; 3 = FAT32
17953 00011C87 771A                <1>       ja    short uflmdt_8
17954                              <1>
17955 00011C89 89F8                <1>       mov   eax, edi ; directory entry index number
17956 00011C8B 66C1E804            <1>       shr   ax, 4 ; 16 entries per sector
17957 00011C8F 034664              <1>       add   eax, [esi+LD_ROOTBegin]
17958                              <1>       ; eax = root directory sector
17959                              <1> uflmdt_6:
17960 00011C92 50                  <1>       push  eax ; * ; disk sector address
17961 00011C93 51                  <1>       push  ecx ; first cluster
17962 00011C94 B901000000          <1>       mov   ecx, 1
17963                              <1>       ; ecx = sector count
17964 00011C99 E8FD080000          <1>       call  disk_read
17965 00011C9E 59                  <1>       pop   ecx
17966 00011C9F 731A                <1>       jnc   short uflmdt_10
17967 00011CA1 58                  <1>       pop   eax ; *
17968                              <1> uflmdt_7:
17969 00011CA2 C3                  <1>       retn
17970                              <1>
17971                              <1> uflmdt_8:
17972 00011CA3 8B5632              <1>       mov   edx, [esi+LD_BPB+FAT32_RootFClust]
17973                              <1> uflmdt_9:
17974 00011CA6 83FA02              <1>       cmp   edx, 2
17975 00011CA9 72F7                <1>       jb    short uflmdt_7 ; invalid, nothing to do
17976                              <1>
17977 00011CAB 83EA02              <1>       sub   edx, 2
17978 00011CAE 89D0                <1>       mov   eax, edx
17979 00011CB0 0FB65613            <1>       movzx edx, byte [esi+LD_BPB+SecPerClust]
17980 00011CB4 F7E2                <1>       mul   edx
17981 00011CB6 034668              <1>       add   eax, [esi+LD_DATABegin]
17982                              <1>       ; eax = sub directory (data) sector
17983 00011CB9 EBD7                <1>       jmp   short uflmdt_6
17984                              <1>
17985                              <1> uflmdt_10:
17986                              <1>       ; Directory sector buffer is ready here!
17987                              <1>       ; OF_FCLUSTER must be compared/verified
17988                              <1>       ; edi = dir entry index number (<= 2047)
17989 00011CBB 6683E70F            <1>       and   di, 0Fh ; 16 entries per sector
17990 00011CBF 66C1E705            <1>       shl   di, 5 ; dir entry index * 32
17991 00011CC3 81C7[84090300]      <1>       add   edi, rw_buffer
17992                              <1>       ;
17993 00011CC9 F6470B18            <1>       test  byte [edi+DirEntry_Attr], 18h ; Vol & Dir
17994 00011CCD 0F856EFFFFFF        <1>       jnz   uflmdt_2 ; not a valid file !
17995 00011CD3 668B5714            <1>       mov   dx, [edi+DirEntry_FstClusHI]
17996 00011CD7 C1E210              <1>       shl   edx, 16
17997 00011CDA 668B571A            <1>       mov   dx, [edi+DirEntry_FstClusLO]
17998 00011CDE 39CA                <1>       cmp   edx, ecx ; same first cluster ?
17999 00011CE0 0F855BFFFFFF        <1>       jne   uflmdt_2 ; no !?
18000                              <1>
18001                              <1>       ; Update directory entry
18002 00011CE6 E80198FFFF          <1>       call  convert_current_date_time
18003                              <1>       ; OUTPUT -> DX = Date in dos dir entry format
18004                              <1>       ;           AX = Time in dos dir entry format
18005 00011CEB 66894716            <1>       mov   [edi+DirEntry_WrtTime], ax
18006 00011CEF 66895718            <1>       mov   [edi+DirEntry_WrtDate], dx
18007 00011CF3 66895712            <1>       mov   [edi+DirEntry_LastAccDate], dx
18008                              <1>
18009 00011CF7 58                  <1>       pop   eax ; *
18010                              <1>
18011 00011CF8 BB[84090300]        <1>       mov   ebx, rw_buffer ; Common r/w sector buffer addr
18012 00011CFD B901000000          <1>       mov   ecx, 1
18013                              <1>       ; esi = logical dos description table address
18014                              <1>       ; eax = disk sector number/address (LBA)
18015                              <1>       ; ecx = sector count
18016                              <1>       ; ebx = buffer address
18017 00011D02 E885080000          <1>       call  disk_write
18018 00011D07 0F8234FFFFFF        <1>       jc    uflmdt_2
18019                              <1>
18020                              <1>       ; save directory buffer if has modified/changed sign
18021 00011D0D E87898FFFF          <1>       call  save_directory_buffer
18022 00011D12 C3                  <1>       retn
18023                              <1>
18024                              <1> uflmdt_11:
18025                              <1>       ; 24/10/2016
18026                              <1>       ; Update last modification date & time of a file
18027                              <1>       ; on a disk with Singlix File System.
18028                              <1>       ;
18029                              <1>       ; (Method: Read the FDT -File Description Table-
18030                              <1>       ; sector of the file and update the lmdt data fields,
18031                              <1>       ; then write FDT sector to the disk.
18032                              <1>       ; /// It is easy but there is compatibility buffer
18033                              <1>       ; method also for changing directory entry data and
18034                              <1>       ; also there are some programming issues for Singlix
18035                              <1>       ; file system (TRFS), which are not completed yet!)
18036                              <1>       ;
18037                              <1>       ; Not ready yet ! (24/10/2016)
18038                              <1>       ; /// Temporary code for error return ! ///
18039 00011D13 31C0                <1>       xor   eax, eax
18040 00011D15 F9                  <1>       stc
18041 00011D16 C3                  <1>       retn
18042                              <1>
18043                              <1> sysalloc:
18044                              <1>       ; 14/10/2017
18045                              <1>       ; 20/08/2017, 01/09/2017
18046                              <1>       ; 20/02/2017, 04/03/2017, 15/05/2017
18047                              <1>       ; 19/02/2017 - TRDOS 386 (TRDOS v2.0)
18048                              <1>       ; (TRDOS 386 feature only!)
18049                              <1>       ;
18050                              <1>       ; Allocate Contiguous Memory Block/Pages (for user)
18051                              <1>       ; (System call for DMA Buffer allocation etc.)
```

```
18052                                  <1>       ;
18053                                  <1>       ; INPUT ->
18054                                  <1>       ;     EBX = Virtual address (for user)
18055                                  <1>       ;          (Physical memory block/aperture
18056                                  <1>       ;          will be mapped to this virtual address)
18057                                  <1>       ;     ECX = Byte Count
18058                                  <1>       ;          (will be rounded up to page border)
18059                                  <1>       ;     If ECX = 0
18060                                  <1>       ;         System call will return with an error (cf=1)
18061                                  <1>       ;         but ECX will contain maximum size of
18062                                  <1>       ;         available memory aperture and physical
18063                                  <1>       ;         (beginning) address of that aperture
18064                                  <1>       ;         (which have maximum size) will be in EAX.
18065                                  <1>       ;     EDX = Upper limit of the requested physical memory
18066                                  <1>       ;           block/pages.
18067                                  <1>       ;           (The last byte address of the memory aperture
18068                                  <1>       ;           must not be equal to or above this limit.)
18069                                  <1>       ;     If EDX = 0
18070                                  <1>       ;        there is NOLIMIT !
18071                                  <1>       ;     If EDX = 0FFFFFFFFh (-1)
18072                                  <1>       ;        ESI = Lower Limit !
18073                                  <1>       ;            (Beginning of the block must not be 'less'
18074                                  <1>       ;            than this.) (Must be equal to or above...)
18075                                  <1>       ;        EDI = Upper Limit !
18076                                  <1>       ;            (End of the block must be !less! than this)
18077                                  <1>       ;            (The last byte addr of the memory aperture
18078                                  <1>       ;            must not be equal to or above this limit.)
18079                                  <1>       ;
18080                                  <1>       ; OUTPUT ->
18081                                  <1>       ;     If CF = 0
18082                                  <1>       ;     EAX = Physical address of the allocated memory block
18083                                  <1>       ;     ECX = Allocated bytes (as rounded up to page borders)
18084                                  <1>       ;     EBX = Virtual address (as rounded up)
18085                                  <1>       ;     If CF = 1
18086                                  <1>       ;         Requested (size of) Memory block could not be
18087                                  <1>       ;         allocated to the user!
18088                                  <1>       ;     If CF = 1 & EAX = 0 (Insufficient memory error!)
18089                                  <1>       ;        ECX = Total number of free bytes
18090                                  <1>       ;              (not size of available contiguous bytes!)
18091                                  <1>       ;     If CF = 1 & EAX > 0
18092                                  <1>       ;        there is not a memory aperture with requested size
18093                                  <1>       ;        but total free mem is not less than requested size.
18094                                  <1>       ;        EAX = Physical addr of available memory aperture
18095                                  <1>       ;             with max size
18096                                  <1>       ;             (but it doesn't fit to the conditions!)
18097                                  <1>       ;        ECX = Size of available memory aperture in bytes.
18098                                  <1>       ;     If CF = 1 -> EAX = 0FFFFFFFFh
18099                                  <1>       ;        Conditions/Parameters are wrong !
18100                                  <1>       ;        ECX is same with input value.
18101                                  <1>       ;
18102                                  <1>       ; Note:      Previously allocated pages will be deallocated if
18103                                  <1>       ;       new allocation conditions are met.
18104                                  <1>       ;
18105                                  <1>       ; Note: u.break control may be included in future versions
18106                                  <1>       ;
18107                                  <1>
18108 00011D17 31C0                   <1>       xor    eax, eax ; 0
18109                                  <1>       ; 14/10/2017
18110 00011D19 4A                     <1>       dec    edx ; is there a limit ?
18111 00011D1A 7810                   <1>       js     short sysalloc_1 ;  0 -> 0FFFFFFFFh -> NO LIMIT
18112 00011D1C 42                     <1>       inc    edx ; > 0
18113                                  <1>       ; Check upper address limit
18114                                  <1>       ;(round up to page borders)
18115 00011D1D 81C1FF0F0000           <1>       add    ecx, PAGE_SIZE-1 ; 4095
18116 00011D23 6681E100F0             <1>       and    cx, ~PAGE_OFF ; not 4095
18117 00011D28 39CA                   <1>       cmp    edx, ecx ; upper limit - block size
18118 00011D2A 7224                   <1>       jb     short sysalloc_err
18119                                  <1> sysalloc_1:
18120                                  <1>       ; EAX = Beginning address (physical)
18121                                  <1>       ; EAX = 0 -> Allocate mem block from the 1st proper aperture
18122                                  <1>       ; ECX = Number of bytes to be allocated
18123 00011D2C E86642FFFF             <1>       call   allocate_memory_block
18124 00011D31 721D                   <1>       jc     short sysalloc_err
18125                                  <1>       ; 01/09/2017
18126 00011D33 29C2                   <1>       sub    edx, eax ; upper limit address - beginning address
18127 00011D35 760F                   <1>       jna    short sysalloc_3 ; begin addr not less than the limit
18128 00011D37 39CA                   <1>       cmp    edx, ecx
18129 00011D39 720B                   <1>       jb     short sysalloc_3 ; end address overs the limit
18130                                  <1> sysalloc_2:
18131                                  <1>       ; EAX = Beginning (physical) addr of the allocated mem block
18132                                  <1>       ; ECX = Num of allocated bytes (rounded up to page borders)
18133 00011D3B 50                     <1>       push   eax ; * ; 04/03/2017
18134                                  <1>       ; Here, requested contiquous memory pages have been allocated
18135                                  <1>       ; on Memory Allocation Table but user's page directory
18136                                  <1>       ; and page tables have not been updated yet!
18137 00011D3C 51                     <1>       push   ecx ; **
18138                                  <1>       ; ebx = virtual address (will be rounded up to page border)
18139                                  <1>       ; ecx = number of bytes to be deallocated
18140                                  <1>       ;      will be adjusted to ebx+ecx round down - ebx round up
18141 00011D3D E8A045FFFF             <1>       call   deallocate_user_pages
18142 00011D42 731F                   <1>       jnc    short sysalloc_4 ; EAX = Deallocated memory bytes
18143 00011D44 59                     <1>       pop    ecx ; **
18144 00011D45 58                     <1>       pop    eax ; *
18145                                  <1> sysalloc_3:
18146                                  <1>       ; error !
18147                                  <1>       ; restore Memory Allocation Table Content
18148 00011D46 E85944FFFF             <1>       call   deallocate_memory_block
18149 00011D4B 31C0                   <1>       xor    eax, eax ; 0
18150 00011D4D 48                     <1>       dec    eax ; 0FFFFFFFFh ; 15/05/2017
18151 00011D4E EB09                   <1>       jmp    short sysalloc_wrong
18152                                  <1> sysalloc_err:
18153 00011D50 8B2D[60030300]         <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
18154 00011D56 894D18                 <1>       mov    [ebp+24], ecx ; return to user with ecx value
18155                                  <1> sysalloc_wrong:
18156                                  <1>       ; eax = 0FFFFFFFFh
```

```
18157 00011D59 A3[64030300]        <1>       mov   [u.r0], eax
18158 00011D5E E94DB6FFFF          <1>       jmp   error
18159                              <1> sysalloc_4:
18160 00011D63 8B2D[60030300]      <1>       mov   ebp, [u.usp]  ; ebp points to user's registers
18161 00011D69 894518              <1>       mov   [ebp+24], eax ; return to user with ecx value
18162 00011D6C 895D10              <1>       mov   [ebp+16], ebx ; new value of ebx (rounded up)
18163 00011D6F 89C1                <1>       mov   ecx, eax ; byte count (from 'deallocate_user_pages')
18164 00011D71 5A                  <1>       pop   edx ; ** ; discard (another) byte count
18165 00011D72 58                  <1>       pop   eax ; *
18166 00011D73 A3[64030300]        <1>       mov   [u.r0], eax ; physical address
18167                              <1>
18168 00011D78 51                  <1>       push  ecx ; 20/08/2017
18169                              <1>       ;
18170                              <1>       ; Write newly allocated contiguous (physical) pages
18171                              <1>       ; on page dir and page tables of current user/process
18172                              <1>       ; as PRESENT, USER, WRITABLE
18173                              <1>       ; (then clear allocated pages)
18174 00011D79 E84C46FFFF          <1>       call  allocate_user_pages
18175                              <1>       ;jnc  sysret ; OK! return to process with success...
18176                              <1>
18177                              <1>       ; 20/08/2017 ('sysdma' modification)
18178 00011D7E 59                  <1>       pop   ecx
18179 00011D7F A1[64030300]        <1>       mov   eax, [u.r0]   ; physical address (of the block)
18180                              <1>
18181 00011D84 721D                <1>       jc    short sysalloc_6
18182                              <1>
18183 00011D86 833D[1C940100]FF    <1>       cmp   dword [dma_addr], 0FFFFFFFFh ; -1
18184 00011D8D 0F823DB6FFFF        <1>       jb    sysret
18185                              <1>
18186 00011D93 A3[1C940100]        <1>       mov   [dma_addr], eax ; save dma address for sysdma
18187 00011D98 890D[20940100]      <1>       mov   [dma_size], ecx ; save dma buff size for sysdma
18188                              <1>
18189 00011D9E E92DB6FFFF          <1>       jmp   sysret
18190                              <1>
18191                              <1> sysalloc_6:
18192                              <1>       ;
18193                              <1>       ; unexpected error ! insufficient memory !? conflict !?
18194                              <1>       ; (!!?there is not a free page for a new page table?!!)
18195                              <1>       ; We need to terminate process with error message !!!
18196                              <1>       ;
18197 00011DA3 8B2D[60030300]      <1>       mov   ebp, [u.usp]  ; ebp points to user's registers
18198 00011DA9 8B4D18              <1>       mov   ecx, [ebp+24] ; byte count
18199                              <1>
18200                              <1>       ; 20/08/2017
18201                              <1>       ;mov  eax, [u.r0]   ; physical address (of the block)
18202                              <1>
18203                              <1>       ;
18204                              <1>       ; restore Memory Allocation Table Content
18205 00011DAC E8F343FFFF          <1>       call  deallocate_memory_block
18206                              <1>       ;
18207 00011DB1 803D[BA6A0000]03    <1>       cmp   byte [CRT_MODE], 3 ; 80x25 text mode?
18208 00011DB8 7407                <1>       je    short sysalloc_7 ; yes
18209                              <1>       ; Current mode is VGA (or CGA graphics) mode,
18210                              <1>       ; We need to return to text mode for displaying
18211                              <1>       ; error message just before 'sysexit'.
18212 00011DBA B003                <1>       mov   al, 3
18213 00011DBC E839FDFEFF          <1>       call  _set_mode
18214                              <1> sysalloc_7:
18215 00011DC1 BE[B63C0100]        <1>       mov   esi, beep_Insufficient_Memory ; error message
18216 00011DC6 E87A52FFFF          <1>       call  print_msg ; print/display the message
18217 00011DCB B801000000          <1>       mov   eax, 1 ; ax=1 is needed for 'sysexit' procedure
18218 00011DD0 E982B7FFFF          <1>       jmp   sysexit ; and terminate the process !
18219                              <1>
18220                              <1> sysdalloc:
18221                              <1>       ; 19/02/2017 - TRDOS 386 (TRDOS v2.0)
18222                              <1>       ; (TRDOS 386 feature only!)
18223                              <1>       ;
18224                              <1>       ; Deallocate Memory Block/Pages (for user)
18225                              <1>       ; (Complementary call for sysalloc.)
18226                              <1>       ;
18227                              <1>       ; INPUT ->
18228                              <1>       ;     EBX = Virtual address (for user)
18229                              <1>       ;           (will be rounded up to page border)
18230                              <1>       ;     ECX = Byte Count
18231                              <1>       ;          (will be adjusted to page borders)
18232                              <1>       ;     If ICX = 0
18233                              <1>       ;        nothing to do
18234                              <1>       ;     If EBX + ECX > User's ESP
18235                              <1>       ;        nothing to do
18236                              <1>       ;
18237                              <1>       ; Note: u.break control may be included in future versions
18238                              <1>       ;
18239                              <1>       ; OUTPUT ->
18240                              <1>       ;     If CF = 0
18241                              <1>       ;        EAX = Deallocated memory bytes
18242                              <1>       ;        EBX = Virtual address (as rounded up)
18243                              <1>       ;     IF CF = 1
18244                              <1>       ;        EAX = 0
18245                              <1>       ;
18246                              <1>       ; Note:      Main purpose of this call is to deallocate/release
18247                              <1>       ;     previously allocated (physically) contiguous memory
18248                              <1>       ;     pages but beginning (virtual) address may not be
18249                              <1>       ;     followed by physically contiguous pages. So, this
18250                              <1>       ;     system call will deallocate user's virtually
18251                              <1>       ;     contiguous memory pages. Also, there is not any
18252                              <1>       ;     objections to use this system call without sysalloc
18253                              <1>       ;     system call; only possible objection is to lost data
18254                              <1>       ;     within user's memory space, if the beginning address
18255                              <1>       ;     and size is not proper.
18256                              <1>       ;
18257                              <1>       ; Note: Empty page tables will not be deallocated!!!
18258                              <1>       ;       (they will be deallocated at process termination)
18259                              <1>       ;
18260                              <1>       ; Note: When the program terminates itself or when it is
18261                              <1>       ;     terminated by operating system kernel, all allocated
```

```
18262                                  <1>       ;      memory pages will be deallocated during termination
18263                                  <1>       ;      stage. So, 'sysdalloc' is not necessary except
18264                                  <1>       ;      forgiving memory block to other programs/processes.
18265                                  <1>       ;
18266 00011DD5 8B15[5C030300]          <1>       mov    edx, [u.sp]
18267 00011DDB 8B420C                  <1>       mov    eax, [edx+12] ; user's stack pointer
18268 00011DDE 29C8                    <1>       sub    eax, ecx ; esp - byte count
18269 00011DE0 24FC                    <1>       and    al, 0FCh ; dword alignment
18270 00011DE2 39D8                    <1>       cmp    eax, ebx
18271 00011DE4 7220                    <1>       jb     short sysdalloc_err ; deallocation overlaps with stack
18272                                  <1>
18273 00011DE6 31C0                    <1>       xor    eax, eax
18274 00011DE8 21C9                    <1>       and    ecx, ecx
18275 00011DEA 7407                    <1>       jz     short sysdalloc_2
18276                                  <1>
18277 00011DEC E8F144FFFF              <1>       call   deallocate_user_pages
18278 00011DF1 7213                    <1>       jc     short sysdalloc_err
18279                                  <1>
18280                                  <1> sysdalloc_2:
18281 00011DF3 A3[64030300]            <1>       mov    [u.r0], eax
18282 00011DF8 8B2D[60030300]          <1>       mov    ebp, [u.usp]
18283 00011DFE 895D10                  <1>       mov    [ebp+16], ebx ; new value of ebx
18284 00011E01 E9CAB5FFFF              <1>       jmp    sysret
18285                                  <1>
18286                                  <1> sysdalloc_err:
18287 00011E06 A3[64030300]            <1>       mov    [u.r0], eax ; 0
18288 00011E0B E9A0B5FFFF              <1>       jmp    error
18289                                  <1>
18290                                  <1> syscalbac:
18291                                  <1>       ; SYS CALLBACK
18292                                  <1>       ; 03/08/2020
18293                                  <1>       ; 16/04/2017
18294                                  <1>       ; 14/04/2017
18295                                  <1>       ; 13/04/2017
18296                                  <1>       ; 28/02/2017
18297                                  <1>       ; 26/02/2017
18298                                  <1>       ; 24/02/2017
18299                                  <1>       ; 21/02/2017 - TRDOS 386 (TRDOS v2.0)
18300                                  <1>       ; (TRDOS 386 feature only!)
18301                                  <1>       ;
18302                                  <1>       ; Link or unlink IRQ callback service to/from user (ring 3)
18303                                  <1>       ;
18304                                  <1>       ; INPUT ->
18305                                  <1>       ;     BL = IRQ number (Hardware interrupt request number)
18306                                  <1>       ;          (0 t0 15 but IRQ 0,1,2,6,8,14,15 are prohibited)
18307                                  <1>       ;          IRQ numbers 3,4,5,7,9,10,11,12,13 are valid
18308                                  <1>       ;          (numbers >15 are invalid)
18309                                  <1>       ;
18310                                  <1>       ;     BH = 0 = Unlink IRQ (in BL) from user (ring 3) service
18311                                  <1>       ;          1 = Link IRQ by using Signal Response Byte method
18312                                  <1>       ;          2 = Link IRQ by using Callback service method
18313                                  <1>       ;          3 = Link IRQ by using Auto Increment S.R.B. method
18314                                  <1>       ;         >3 = invalid
18315                                  <1>       ;
18316                                  <1>       ;     CL = Signal Return/Response Byte value
18317                                  <1>       ;
18318                                  <1>       ;     If BH = 3, kernel will put a counter value  ; 03/08/2020
18319                                  <1>       ;               (into the S.R.B. addr)
18320                                  <1>       ;             between 0 to 255. (start value = CL+1)
18321                                  <1>       ;
18322                                  <1>       ;     NOTE: counter value, for example: even and odd numbers
18323                                  <1>       ;           may be used for -audio- DMA buffer switch
18324                                  <1>       ;           within double buffer method, etc.
18325                                  <1>       ;
18326                                  <1>       ;     EDX = Signal return (Response) byte address
18327                                  <1>       ;                           - or -
18328                                  <1>       ;           Interrupt/Callback service/routine address
18329                                  <1>       ;
18330                                  <1>       ;           (virtual address in user's memory space)
18331                                  <1>       ;
18332                                  <1>       ; OUTPUT ->
18333                                  <1>       ;     CF = 0 & EAX = 0 -> Successful setting
18334                                  <1>       ;     CF = 1 & EAX > 0 -> IRQ is prohibited or locked
18335                                  <1>       ;                 by another process
18336                                  <1>       ;            eax = ERR_PERM_DENIED -> prohibited or locked
18337                                  <1>       ;            eax = ERR_INV_PARAMETER ->
18338                                  <1>       ;                  invalid parameter/option or bad address
18339                                  <1>       ;
18340                                  <1>       ;     NOTE: Timer callbacks are set by using 'systimer'
18341                                  <1>       ;           system call (IRQ 0, PIT and IRQ 8, RTC)
18342                                  <1>       ;
18343                                  <1>       ;           Direct keyboard access is performed by using
18344                                  <1>       ;           Keyboard Interrupt (INT 32h)
18345                                  <1>       ;
18346                                  <1>       ;           It is prohibited here because:
18347                                  <1>       ;            1) Signal Response Byte method has not advantage
18348                                  <1>       ;               against INT 32h, function AH = 1. Also,
18349                                  <1>       ;               keyboard service interrupt will return with
18350                                  <1>       ;               ascii and scan codes (AL, AH) while
18351                                  <1>       ;               SRB method has only 1 byte space for ascii code
18352                                  <1>       ;               or scan code. One byte signal response is used
18353                                  <1>       ;               for ensuring very simple and very fast
18354                                  <1>       ;               virtual to physical memory address conversion
18355                                  <1>       ;               without any memory page crossover risk.
18356                                  <1>       ;               (Otherwise double page conversion or word
18357                                  <1>       ;               alignment would be needed.)
18358                                  <1>       ;            2) Badly written user code (callback code)
18359                                  <1>       ;               can prevent keyboard and timesharing functions
18360                                  <1>       ;               of the operating system via continuous and long
18361                                  <1>       ;               keyboard event handling by callback service.
18362                                  <1>       ;               (It can cause to lose immediate keystroke
18363                                  <1>       ;               response from hardware to user.)
18364                                  <1>       ;            3) If user will check any keyboard events, 'getkey'
18365                                  <1>       ;               (or 'getchar') must have more priority than other
18366                                  <1>       ;               (video etc.) events because only control ability
```

```
18367                                  <1>       ;               on a procedural infinite loop is a keyboard or
18368                                  <1>       ;               mouse event. So user can use keyboard function
18369                                  <1>       ;               at the end or at the beginning of a loop.
18370                                  <1>       ;               In this case, INT 32h is used for that purpose
18371                                  <1>       ;               and timer interrupt etc. callbacks can be used
18372                                  <1>       ;               for dynamic and synchronized data refresh/transfer
18373                                  <1>       ;               while cpu is in a static loop (without polling).
18374                                  <1>       ;               Keyboard Int callback is not more useful because
18375                                  <1>       ;               already a manual check (a key is pressed or not)
18376                                  <1>       ;               can be performed (via INT 32h, AH = 1) efficiently
18377                                  <1>       ;               in a loop to prevent a locked infinitive loop.
18378                                  <1>       ;
18379                                  <1>       ;         Disk IRQs (6,14,15) have been phohibited from ring 3
18380                                  <1>       ;         callback because, disk operations (file system services
18381                                  <1>       ;         etc.) are independent from user program, for fast disk r/w.
18382                                  <1>       ;         They are not more useful at ring 3 while they are in use
18383                                  <1>       ;         by standard diskio functions which are mandatory part of
18384                                  <1>       ;         (monolithic) OS kernel and mainprog command interpreter.
18385                                  <1>       ;         INT 33h diskio functions are enough for user level disk
18386                                  <1>       ;         r/w.
18387                                  <1>       ;
18388                                  <1>       ; TRDOS 386 - IRQ CALLBACK structures (parameters):
18389                                  <1>       ;
18390                                  <1>       ;        [u.irqlock] = 1 word, IRQ flags (0-15) that indicates
18391                                  <1>       ;                  which IRQs are locked by (that) user.
18392                                  <1>       ;                    Lock and unlock (by user) will change
18393                                  <1>       ;                  these flags or 'terminate process' (sysexit)
18394                                  <1>       ;                  will clear these flags and unlock those IRQs.
18395                                  <1>       ;
18396                                  <1>       ;                  Bit 0 is for IRQ 0 and Bit 15 is for IRQ 15
18397                                  <1>       ;
18398                                  <1>       ;        IRQ(x).owner      : 1 byte, user, [u.uno], 0 = free (unlocked)
18399                                  <1>       ;
18400                                  <1>       ;        IRQ(x).method : 1 byte for callback method & status
18401                                  <1>       ;                     0 = Signal Response Byte method
18402                                  <1>       ;                     1 = Callback service method
18403                                  <1>       ;                     >1 = invalid for current 'syscalback'.
18404                                  <1>       ;                   or(+) 80h = IRQ is in use by system (ring 0)
18405                                  <1>       ;                               function (audio etc.) or
18406                                  <1>       ;                             a device driver.
18407                                  <1>       ;                   (system function will ignore the lock/owner)
18408                                  <1>       ;
18409                                  <1>       ;        IRQ(x).srb: 1 byte, Signal Return/Response byte value
18410                                  <1>       ;                    (a fixed value by user or a counter value
18411                                  <1>       ;                   from 0 to 255, which is increased by every
18412                                  <1>       ;                   interrupt just before putting it into
18413                                  <1>       ;                   the Signal Response byte address
18414                                  <1>       ;                   (This is not used in callback serv method)
18415                                  <1>       ;
18416                                  <1>       ;        IRQ(x).addr      : 1 dword
18417                                  <1>       ;                    Signal Response Byte address (physical)
18418                                  <1>       ;                                -or-
18419                                  <1>       ;                    Callback service address (virtual)
18420                                  <1>       ;
18421                                  <1>       ;        IRQ(x).dev: 1 byte
18422                                  <1>       ;                    0 = Default device or kernel function
18423                                  <1>       ;                                -or-
18424                                  <1>       ;                    1-255 = Assigned device driver number
18425                                  <1>       ;
18426                                  <1>       ;        (x) = 3,4,5,7,9,10,11,12,13
18427                                  <1>       ;
18428                                  <1>       ;
18429                                  <1>       ;     NOTE: If user's process/program calls the kernel (INT 40h)
18430                                  <1>       ;           while it is already running in a (ring 3) callback
18431                                  <1>       ;           service, kernel will force (convert) system call to
18432                                  <1>       ;           'sysrele' (sys release). So, this feature provides
18433                                  <1>       ;           easy and simple usage of callback services without
18434                                  <1>       ;           falling into deepless <please 'callback me' then
18435                                  <1>       ;           let me 'callback you'> cycles! (User must return
18436                                  <1>       ;           from callback service by using 'sysrele' system
18437                                  <1>       ;           call, without a significant delay. Otherwise user
18438                                  <1>       ;           process/program may be late to catch the next event
18439                                  <1>       ;           within same callback purpose.
18440                                  <1>       ;
18441                                  <1>
18442 00011E10 30C0                   <1>       xor    al, al ; the caller is 'syscalbac' sign/flag
18443 00011E12 E813170000             <1>       call   set_irq_callback_service
18444                                  <1>       ; 16/04/2017
18445 00011E17 A3[64030300]           <1>       mov    [u.r0], eax
18446 00011E1C 0F83AEB5FFFF           <1>       jnc    sysret
18447 00011E22 A3[C8030300]           <1>       mov    dword [u.error], eax
18448 00011E27 E984B5FFFF             <1>       jmp    error
18449                                  <1>
18450                                  <1> sysfpstat:
18451                                  <1>       ; 28/02/2017 - TRDOS 386 (TRDOS v2.0)
18452                                  <1>       ; (TRDOS 386 feature only!)
18453                                  <1>       ;
18454                                  <1>       ; Set or reset FPU registers save/restore option (for user)
18455                                  <1>       ;             (during software task switching, wswap-rswap)
18456                                  <1>       ;
18457                                  <1>       ; INPUT ->
18458                                  <1>       ;     BL = 0 -> reset
18459                                  <1>       ;     BL = 1 -> set (FPU register will be saved and restored)
18460                                  <1>       ;
18461                                  <1>       ; OUTPUT ->
18462                                  <1>       ;     cf = 0 -> no error, FPU is ready...
18463                                  <1>       ;               (EAX = 0)
18464                                  <1>       ;     Cf = 1 -> error, 80387 FPU is not ready !
18465                                  <1>       ;               (EAX = 0FFFFFFFFh)
18466                                  <1>
18467 00011E2C 31C0                   <1>       xor    eax, eax
18468 00011E2E 803D[648D0100]00       <1>       cmp    byte [fpready], 0
18469 00011E35 7613                   <1>       jna    short sysfpstat_err
18470                                  <1>
18471 00011E37 80E301                 <1>       and    bl, 1 ; use BIT 0 only !
```

```
18472 00011E3A 881D[DA030300]      <1>       mov    [u.fpsave], bl
18473 00011E40 A3[64030300]        <1>       mov    [u.r0], eax ; 0
18474 00011E45 E986B5FFFF          <1>       jmp    sysret
18475                              <1>
18476                              <1> sysfpstat_err:
18477 00011E4A 48                  <1>       dec    eax ; 0FFFFFFFFh
18478 00011E4B A3[64030300]        <1>       mov    [u.r0], eax ; -1
18479 00011E50 E95BB5FFFF          <1>       jmp    error
18480                              <1>
18481                              <1> sysdelete: ; Delete (Remove, Unlink) File
18482                              <1>       ; 29/12/2017 (TRDOS 386 = TRDOS v2.0)
18483                              <1>       ;
18484                              <1>         ; INPUT ->
18485                              <1>         ;          EBX = File name (ASCIIZ string) address
18486                              <1>       ; OUTPUT ->
18487                              <1>       ;          cf = 0 -> eax = 0
18488                              <1>       ;          cf = 1 -> Error code in AL
18489                              <1>       ;
18490                              <1>       ; Modified Registers: EAX (at the return of system call)
18491                              <1>       ;
18492                              <1>
18493 00011E55 89DE                <1>       mov    esi, ebx
18494                              <1>       ; file name is forced, change directory as temporary
18495                              <1>       ;mov   ax, 1
18496                              <1>       ;mov   [FFF_Valid], ah ; 0 ; reset
18497                              <1>       ;call  set_working_path
18498 00011E57 E8620B0000          <1>       call   set_working_path_x
18499 00011E5C 731D                <1>       jnc    short sysdelete_1
18500                              <1>
18501 00011E5E 21C0                <1>       and    eax, eax  ; 0 -> Bad Path!
18502 00011E60 7505                <1>       jnz    short sysdelete_err
18503                              <1>       ; eax = 0
18504                              <1> sysdelete_path_err:
18505 00011E62 B813000000          <1>       mov    eax, ERR_INV_PATH_NAME ; 'bad path name !'
18506                              <1> sysdelete_err:
18507 00011E67 A3[64030300]        <1>       mov    [u.r0], eax
18508 00011E6C A3[C8030300]        <1>       mov    [u.error], eax
18509 00011E71 E81D0C0000          <1>       call   reset_working_path
18510 00011E76 E935B5FFFF          <1>       jmp    error
18511                              <1> sysdelete_1:
18512                              <1>       ;mov   esi, FindFile_Name
18513 00011E7B 66B80018            <1>       mov    ax, 1800h ; Only files
18514 00011E7F E8D170FFFF          <1>       call   find_first_file
18515 00011E84 72E1                <1>       jc     short sysdelete_err
18516                              <1> sysdelete_2:
18517                              <1>       ; check file attributes
18518                              <1>
18519                              <1>       ;test  bl, 17 ; system, hidden, readonly, directory
18520 00011E86 F6C307              <1>       test   bl, 7 ; system, hidden, readonly
18521 00011E89 7407                <1>       jz     short sysdelete_3
18522                              <1>
18523 00011E8B B80B000000          <1>       mov    eax, ERR_FILE_ACCESS ; 11 = 'permission denied !'
18524 00011E90 EBD5                <1>       jmp    short sysdelete_err
18525                              <1> sysdelete_3:
18526 00011E92 6621D2              <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
18527 00011E95 7407                <1>       jz     short sysdelete_4
18528 00011E97 B81A000000          <1>       mov    eax, ERR_INV_FILE_NAME ; 26 = 'invalid file name !'
18529 00011E9C EBC9                <1>       jmp    short sysdelete_err
18530                              <1> sysdelete_4:
18531                              <1>       ;mov   bh, [LongName_EntryLength]
18532 00011E9E 883D[D68A0100]      <1>       mov    [DelFile_LNEL], bh ; Long name entry length (if > 0)
18533                              <1>       ; edi = Directory Entry Offset (DirBuff)
18534                              <1>       ; esi = Directory Entry (FFF Structure)
18535 00011EA4 E8299AFFFF          <1>       call   remove_file
18536 00011EA9 72BC                <1>       jc     short sysdelete_err
18537                              <1> sysrmdir_5:
18538 00011EAB 31C0                <1>       xor    eax, eax ; 0
18539 00011EAD A3[64030300]        <1>       mov    [u.r0], eax
18540                              <1>       ;mov   [u.error], eax
18541 00011EB2 E8DC0B0000          <1>       call   reset_working_path
18542 00011EB7 E914B5FFFF          <1>       jmp    sysret
18543                              <1>
18544                              <1>
18545                              <1> sysrmdir: ; Remove (Unlink) Directory
18546                              <1>       ; 19/01/2021
18547                              <1>       ; 29/12/2017 (TRDOS 386 = TRDOS v2.0)
18548                              <1>       ;
18549                              <1>         ; INPUT ->
18550                              <1>         ;          EBX = Pointer to directory name
18551                              <1>       ; OUTPUT ->
18552                              <1>       ;          cf = 0 -> eax = 0
18553                              <1>       ;          cf = 1 -> Error code in AL
18554                              <1>       ;
18555                              <1>       ; Modified Registers: EAX (at the return of system call)
18556                              <1>       ;
18557                              <1>
18558                              <1>       ; 19/01/2021
18559 00011EBC 803D[B0030300]00    <1>       cmp    byte [u.uid], 0  ; root (super user) ?
18560 00011EC3 7614                <1>       jna    short sysrmdir_0
18561                              <1>
18562                              <1>       ;mov   dword [u.r0], ERR_PERM_DENIED
18563 00011EC5 B80B000000          <1>       mov    eax, ERR_PERM_DENIED ; ERR_NOT_SUPERUSER
18564 00011ECA A3[64030300]        <1>       mov    [u.r0], eax
18565 00011ECF A3[C8030300]        <1>       mov    [u.error], eax
18566 00011ED4 E9D7B4FFFF          <1>       jmp    error
18567                              <1>
18568                              <1> sysrmdir_0:
18569 00011ED9 89DE                <1>       mov    esi, ebx
18570                              <1>       ; file name is forced, change directory as temporary
18571                              <1>       ;mov   ax, 1
18572                              <1>       ;mov   [FFF_Valid], ah ; 0 ; reset
18573                              <1>       ;call  set_working_path
18574 00011EDB E8DE0A0000          <1>       call   set_working_path_x
18575 00011EE0 731D                <1>       jnc    short sysrmdir_1
18576                              <1>
```

```
18577 00011EE2 21C0              <1>       and    eax, eax  ; 0 -> Bad Path!
18578 00011EE4 7505              <1>       jnz    short sysrmdir_err
18579                            <1>       ; eax = 0
18580                            <1> sysrmdir_not_found:
18581 00011EE6 B80C000000        <1>       mov    eax, ERR_DIR_NOT_FOUND ; Directory not found !
18582                            <1> sysrmdir_err:
18583 00011EEB A3[64030300]      <1>       mov    [u.r0], eax
18584 00011EF0 A3[C8030300]      <1>       mov    [u.error], eax
18585 00011EF5 E8990B0000        <1>       call   reset_working_path
18586 00011EFA E9B1B4FFFF        <1>       jmp    error
18587                            <1> sysrmdir_1:
18588                            <1>       ;mov   esi, FindFile_Name
18589 00011EFF 66B81008          <1>       mov    ax, 0810h ; Only directories
18590 00011F03 E84D70FFFF        <1>       call   find_first_file
18591 00011F08 7306              <1>       jnc    short sysrmdir_2
18592                            <1>
18593                            <1>       ; eax = 2 (File not found !)
18594 00011F0A 3C02              <1>       cmp    al, 2 ; ERR_NOT_FOUND
18595 00011F0C 74D8              <1>       je     short sysrmdir_not_found
18596 00011F0E EBDB              <1>       jmp    short sysrmdir_err
18597                            <1> sysrmdir_2:
18598                            <1>       ; check directory attributes
18599                            <1>
18600 00011F10 F6C307            <1>       test   bl, 7 ; system, hidden, readonly
18601 00011F13 7407              <1>       jz     short sysrmdir_3
18602                            <1>
18603 00011F15 B80B000000        <1>       mov    eax, ERR_DIR_ACCESS ; 11 = 'permission denied !'
18604 00011F1A EBCF              <1>       jmp    short sysrmdir_err
18605                            <1> sysrmdir_3:
18606 00011F1C 6621D2            <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
18607 00011F1F 7407              <1>       jz     short sysrmdir_4
18608                            <1>       ;mov   eax, ERR_NOT_DIR ; 'not a valid directory !'
18609 00011F21 B813000000        <1>       mov    eax, ERR_INV_PATH_NAME ; 'bad path name !'
18610 00011F26 EBC3              <1>       jmp    short sysrmdir_err
18611                            <1> sysrmdir_4:
18612                            <1>       ;mov   bh, [LongName_EntryLength]
18613 00011F28 883D[D68A0100]    <1>       mov    [DelFile_LNEL], bh ; Long name entry length (if > 0)
18614                            <1>       ; edi = Directory Entry Offset (DirBuff)
18615                            <1>       ; esi = Directory Entry (FFF Structure)
18616 00011F2E E8F776FFFF        <1>       call   delete_sub_directory
18617 00011F33 0F8372FFFFFF      <1>       jnc    sysrmdir_5
18618                            <1> ;     jc     short sysrmdir_6
18619                            <1> ;
18620                            <1> ;     xor    eax, eax ; 0
18621                            <1> ;sysrmdir_5:
18622                            <1> ;     mov    [u.r0], eax
18623                            <1> ;     ;mov   [u.error], eax
18624                            <1> ;     call   reset_working_path
18625                            <1> ;     jmp    sysret
18626                            <1> sysrmdir_6:
18627 00011F39 A3[64030300]      <1>       mov    [u.r0], eax
18628 00011F3E A3[C8030300]      <1>       mov    [u.error], eax
18629                            <1>
18630 00011F43 09C0              <1>       or     eax, eax ; EAX = 0 -> Directory not empty!
18631 00011F45 741C              <1>       jz     short sysrmdir_9
18632                            <1>
18633                            <1>       ; EAX > 0 -> Error code in AL (or AX or EAX)
18634                            <1>
18635 00011F47 833D[8A880100]01  <1>       cmp    dword [FAT_ClusterCounter], 1
18636 00011F4E 7209              <1>       jb     short sysrmdir_8
18637                            <1> sysrmdir_7:
18638                            <1>       ; ESI = Logical DOS Drive Description Table address
18639 00011F50 66BB00FF          <1>       mov    bx, 0FF00h ; BH = FFh -> use ESI for Drive parameters
18640                            <1>                   ; BL = 0 -> Recalculate free cluster count
18641 00011F54 E85DAFFFFF        <1>       call   calculate_fat_freespace
18642                            <1> sysrmdir_8:
18643 00011F59 E8350B0000        <1>       call   reset_working_path
18644 00011F5E E94DB4FFFF        <1>       jmp    error
18645                            <1>
18646                            <1> sysrmdir_9:
18647 00011F63 A1[8A880100]      <1>       mov    eax, [FAT_ClusterCounter]
18648 00011F68 09C0              <1>       or     eax, eax ; 0 ?
18649 00011F6A 0F847BFFFFFF      <1>       jz     sysrmdir_err
18650                            <1>       ; ESI = Logical DOS Drive Description Table address
18651 00011F70 66BB01FF          <1>       mov    bx, 0FF01h ; BH = FFh -> use ESI for Drive parameters
18652                            <1>                   ; BL = 1 -> add free clusters
18653 00011F74 E83DAFFFFF        <1>       call   calculate_fat_freespace
18654 00011F79 09C9              <1>       or     ecx, ecx
18655 00011F7B 74DC              <1>         jz   short sysrmdir_8 ; ecx = 0 -> OK
18656                            <1>       ; ecx > 0 -> Error (Recalculation is needed)
18657 00011F7D EBD1              <1>       jmp    short sysrmdir_7
18658                            <1>
18659                            <1>
18660                            <1> syschdir: ; Change Current (Working) Drive & Directory (for user)
18661                            <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
18662                            <1>       ;
18663                            <1>         ; INPUT ->
18664                            <1>         ;          EBX = Directory name (ASCIIZ string) address
18665                            <1>       ; OUTPUT ->
18666                            <1>       ;          cf = 0 -> eax = 0
18667                            <1>       ;          cf = 1 -> Error code in AL
18668                            <1>       ;
18669                            <1>       ; Modified Registers: EAX (at the return of system call)
18670                            <1>       ;
18671                            <1>       ; NOTE: If drive name is not included, only the working
18672                            <1>       ; directory (for user, not for drive/OS) will be chanded.
18673                            <1>       ; If there is a drive name (as A:, B:, C:, D: etc.)
18674                            <1>       ; at the beginning of the ASCIIZ (directory) string,
18675                            <1>       ; working drive and working directory (for user)
18676                            <1>       ; will be changed together.
18677                            <1>       ; (When the program is terminated, MainProg -internal
18678                            <1>       ; shell- will reset working directory to the previous
18679                            <1>       ; -current- logical drive's current directory again.)
18680                            <1>
18681 00011F7F 89DE              <1>       mov    esi, ebx
```

```
18682                                  <1>       ; file name is not forced, change directory as temporary
18683 00011F81 31C0                    <1>       xor    eax, eax
18684                                  <1>       ;mov   [FFF_Valid], ah ; 0 ; reset
18685                                  <1>       ;call  set_working_path
18686 00011F83 E83A0A0000              <1>       call   set_working_path_xx
18687 00011F88 731D                    <1>       jnc    short syschdir_ok
18688 00011F8A 21C0                    <1>       and    eax, eax  ; 0 -> Bad Path!
18689 00011F8C 7505                    <1>       jnz    short syschdir_err
18690                                  <1>       ; eax = 0
18691                                  <1> syschdir_not_found:
18692 00011F8E B80C000000              <1>       mov    eax, ERR_DIR_NOT_FOUND ; Directory not found !
18693                                  <1> syschdir_err:
18694 00011F93 A3[64030300]            <1>       mov    [u.r0], eax
18695 00011F98 A3[C8030300]            <1>       mov    [u.error], eax
18696 00011F9D E8F10A0000              <1>       call   reset_working_path
18697 00011FA2 E909B4FFFF              <1>       jmp    error
18698                                  <1> syschdir_ok:
18699 00011FA7 31C0                    <1>       xor    eax, eax ; 0
18700 00011FA9 A3[64030300]            <1>       mov    [u.r0], eax
18701                                  <1>       ;mov   [u.error], eax
18702 00011FAE E91DB4FFFF              <1>       jmp    sysret
18703                                  <1>
18704                                  <1>
18705                                  <1> syschmod: ; Get & Change File (or Directory) Attributes
18706                                  <1>       ; 19/01/2021
18707                                  <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
18708                                  <1>       ;
18709                                  <1>         ; INPUT ->
18710                                  <1>         ;          EBX = File/Directory (ASCIIZ) name address
18711                                  <1>       ;          CL = New attributes (if CL < 40h)
18712                                  <1>       ;          CL >= 40h -> Get File Attributes
18713                                  <1>       ; OUTPUT ->
18714                                  <1>       ;          cf = 0 -> EAX = File attributes (in AL)
18715                                  <1>       ;          cf = 1 -> Error code in AL
18716                                  <1>       ;
18717                                  <1>       ; Modified Registers: EAX (at the return of system call)
18718                                  <1>       ;
18719                                  <1>       ; MSDOS File Attributes:    (bit value of attrib byte)
18720                                  <1>       ;     ATTR_READ_ONLY      =     01h  (bit 0, 'R')
18721                                  <1>       ;     ATTR_HIDDEN  =      02h  (bit 1, 'H')
18722                                  <1>       ;     ATTR_SYSTEM  =      04h  (bit 2, 'S')
18723                                  <1>       ;     ATTR_VOLUME_ID      =     08h  (bit 3)
18724                                  <1>       ;     ATTR_DIRECTORY      =     10h  (bit 4)
18725                                  <1>       ;     ATTR_ARCHIVE =      20h  (bit 5, 'A')
18726                                  <1>       ;     ATTR_LONG_NAME      =     ATTR_READONLY |
18727                                  <1>       ;                           ATTR_HIDDEN |
18728                                  <1>       ;                           ATTR_SYSTEM |
18729                                  <1>       ;                           ATTR_VOLUME_ID
18730                                  <1>       ;     The upper two bits of attributes must be 0.
18731                                  <1>       ;
18732                                  <1>       ; Note:      * If ATTR_DIRECTORY is set, only directory names
18733                                  <1>       ;       will be searched (and S,H,R,A attributeds of
18734                                  <1>       ;       the directory will be changed.)
18735                                  <1>       ;     * If ATTR_VOLUME_ID is set, 'syschmod' system call
18736                                  <1>       ;       will return with 'permission denied' error.
18737                                  <1>       ;     * If ATTR_DIRECTORY is not set, only file names
18738                                  <1>       ;       will be searched (and S,H,R,A attributes of the
18739                                  <1>       ;       file will be changed.)
18740                                  <1>       ;
18741                                  <1>       ; (Ony Super User can change S,H,R attributes.)
18742                                  <1>
18743 00011FB3 80F940                  <1>       cmp    cl, 40h
18744 00011FB6 7327                    <1>       jnb    short syschmod_0
18745                                  <1>
18746 00011FB8 F6C108                  <1>       test   cl, 08h ; ATTR_VOLUME_ID
18747 00011FBB 750E                    <1>       jnz    short syschmod_perm_err
18748                                  <1>
18749                                  <1>       ; 19/01/2021
18750 00011FBD 803D[B0030300]00        <1>       cmp    byte [u.uid], 0  ; root (super user) ?
18751 00011FC4 7619                    <1>       jna    short syschmod_0
18752                                  <1>
18753                                  <1>       ; Not super user..
18754 00011FC6 F6C107                  <1>       test   cl, 07h      ; S,H,R attributes
18755 00011FC9 7414                    <1>       jz     short syschmod_0
18756                                  <1>
18757                                  <1> syschmod_perm_err:
18758                                  <1>       ;mov   dword [u.r0], ERR_PERM_DENIED
18759 00011FCB B80B000000              <1>       mov    eax, ERR_PERM_DENIED ; 'permission denied !'
18760 00011FD0 A3[64030300]            <1>       mov    [u.r0], eax
18761 00011FD5 A3[C8030300]            <1>       mov    [u.error], eax
18762 00011FDA E9D1B3FFFF              <1>       jmp    error
18763                                  <1>
18764                                  <1> syschmod_0:
18765 00011FDF 880D[248B0100]          <1>       mov    [Attributes], cl
18766 00011FE5 89DE                    <1>       mov    esi, ebx
18767                                  <1>       ; file name is forced, change directory as temporary
18768                                  <1>       ;mov   ax, 1
18769                                  <1>       ;mov   [FFF_Valid], ah ; 0 ; reset
18770                                  <1>       ;call  set_working_path
18771 00011FE7 E8D2090000              <1>       call   set_working_path_x
18772 00011FEC 731D                    <1>       jnc    short syschmod_1
18773 00011FEE 21C0                    <1>       and    eax, eax  ; 0 -> Bad Path!
18774 00011FF0 7505                    <1>       jnz    short syschmod_err
18775                                  <1>       ; eax = 0
18776                                  <1> syschmod_path_not_found:
18777 00011FF2 B813000000              <1>       mov    eax, ERR_INV_PATH_NAME ; 'Bad path name !'
18778                                  <1> syschmod_err:
18779 00011FF7 A3[64030300]            <1>       mov    [u.r0], eax
18780 00011FFC A3[C8030300]            <1>       mov    [u.error], eax
18781 00012001 E88D0A0000              <1>       call   reset_working_path
18782 00012006 E9A5B3FFFF              <1>       jmp    error
18783                                  <1> syschmod_1:
18784 0001200B B008                    <1>       mov    al, 08h ; Except volume labels (& long names)
18785 0001200D A0[248B0100]            <1>       mov    al, [Attributes]
18786 00012012 2410                    <1>       and    al, 10h ;
```

```
18787                                  <1>       ;mov   esi, FindFile_Name
18788                                  <1>       ;mov   ax, 1800h ; Only files
18789                                  <1>       ;mov   ax, 0810h ; Only directories
18790 00012014 E83C6FFFFF              <1>       call   find_first_file
18791                                  <1>       ;jnc   short syschmod_2
18792 00012019 72DC                    <1>       jc     short syschmod_err
18793                                  <1>
18794                                  <1>       ;; eax = 2 (File not found !)
18795                                  <1>       ;cmp   al, 2 ; ERR_NOT_FOUND
18796                                  <1>       ;jne   short syschmod_err
18797                                  <1>
18798                                  <1>       ;and   byte [Attributes], 10h
18799                                  <1>       ;jz    short syschmod_err
18800                                  <1>
18801                                  <1>       ;; Directory not found !
18802                                  <1>       ;mov   al, 3 ; ERR_PATH_NOT_FOUND
18803                                  <1>       ;jmp   short syschmod_err
18804                                  <1>
18805                                  <1> syschmod_2:
18806 0001201B 6621D2                  <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
18807 0001201E 7407                    <1>       jz     short syschmod_3
18808 00012020 B81A000000              <1>       mov    eax, ERR_INV_FILE_NAME ; 'invalid file name !'
18809 00012025 EBD0                    <1>       jmp    short syschmod_err
18810                                  <1> syschmod_3:
18811                                  <1>       ; EDI = Directory buffer entry offset/address
18812                                  <1>       ; BL = File (or Directory) Attributes
18813                                  <1>       ; mov  bl, [EDI+0Bh]
18814                                  <1>
18815                                  <1>       ; check directory attributes
18816 00012027 8A3D[248B0100]          <1>       mov    bh, [Attributes] ; new attributes
18817 0001202D 80FF40                  <1>       cmp    bh, 40h  ;>=40 -> get file/directory attributes
18818 00012030 732D                    <1>       jnb    short syschmod_6
18819                                  <1>
18820                                  <1>       ; set file/directory attributes
18821 00012032 F6C307                  <1>       test   bl, 7 ; system, hidden, readonly
18822 00012035 7409                    <1>       jz     short syschmod_4
18823                                  <1>
18824                                  <1>       ; 19/01/2021
18825 00012037 803D[B0030300]00        <1>       cmp    byte [u.uid], 0  ; root (super user) ?
18826 0001203E 778B                    <1>       ja     short syschmod_perm_err
18827                                  <1> syschmod_4:
18828 00012040 66817F0CA101            <1>       cmp    word [edi+DirEntry_NTRes], 01A1h ; Singlix FS
18829 00012046 7424                    <1>       je     short syschmod_7
18830                                  <1>
18831 00012048 887F0B                  <1>       mov    [edi+0Bh], bh    ; Attributes (New!)
18832                                  <1>
18833 0001204B C605[94880100]02        <1>       mov    byte [DirBuff_ValidData], 2 ; modified sign
18834                                  <1>                                       ; to force write
18835 00012052 E83395FFFF              <1>       call   save_directory_buffer
18836 00012057 729E                    <1>       jc     short syschmod_err
18837                                  <1>
18838                                  <1> syschmod_5:
18839 00012059 8A1D[248B0100]          <1>       mov    bl, [Attributes]
18840                                  <1> syschmod_6:
18841 0001205F 0FB6C3                  <1>       movzx  eax, bl
18842 00012062 A3[64030300]            <1>       mov    [u.r0], eax
18843                                  <1>       ;mov   dword [u.error], 0
18844 00012067 E964B3FFFF              <1>       jmp    sysret
18845                                  <1>
18846                                  <1> syschmod_7:
18847 0001206C 29C0                    <1>       sub    eax, eax
18848 0001206E 8A25[92880100]          <1>       mov    ah, [DirBuff_DRV]
18849 00012074 BE00010900              <1>       mov    esi, Logical_DOSDisks
18850 00012079 01C6                    <1>       add    esi, eax
18851 0001207B 807E04A1                <1>       cmp    byte [esi+LD_FSType], 0A1h
18852 0001207F 7307                    <1>       jnc    short syschmod_8
18853 00012081 B01D                    <1>       mov    al, ERR_INV_DATA ; 29 = Invalid Data
18854 00012083 E96FFFFFFF              <1>       jmp    syschmod_err
18855                                  <1>
18856                                  <1> syschmod_8:
18857                                  <1>       ; BH = New MS-DOS File Attributes
18858 00012088 88F8                    <1>       mov    al, bh ; File/Directory Attributes
18859 0001208A 30E4                    <1>       xor    ah, ah ; Attributes in MS-DOS format sign
18860 0001208C E82280FFFF              <1>       call   change_fs_file_attributes
18861 00012091 0F8260FFFFFF            <1>       jc     syschmod_err
18862 00012097 EBC0                    <1>       jmp    short syschmod_5
18863                                  <1>
18864                                  <1>
18865                                  <1> sysdrive: ; Get/Set Current (Working) Drive (for user)
18866                                  <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
18867                                  <1>       ;
18868                                  <1>         ; INPUT ->
18869                                  <1>         ;          BL = Logical DOS Drive number (0=A: ... 2=C:)
18870                                  <1>       ;         If BL = 0FFh -> Get Current Drive
18871                                  <1>       ; OUTPUT ->
18872                                  <1>       ;          cf = 0 ->
18873                                  <1>       ;               AL = Current Drive number
18874                                  <1>       ;               AH = The Last Logical DOS Drive no.
18875                                  <1>       ;          cf = 1 -> Error code in AL
18876                                  <1>       ;
18877                                  <1>       ; Modified Registers: EAX (at the return of system call)
18878                                  <1>       ;
18879                                  <1>       ; NOTE: If the requested logical dos drive is ready,
18880                                  <1>       ;     it's current current directory will be the user's
18881                                  <1>       ;     (program's) current directory.
18882                                  <1>       ;     (When the program is terminated, MainProg -internal
18883                                  <1>       ;     shell- will reset the previous -current- logical drive
18884                                  <1>       ;       as current drive again).
18885                                  <1>
18886 00012099 80FBFF                  <1>       cmp    bl, 0FFh
18887 0001209C 7435                    <1>       je     short sysdrive_ok
18888 0001209E 3A1D[54390100]          <1>       cmp    bl, [Last_DOS_DiskNo]
18889 000120A4 771E                    <1>       ja     short sysdrive_err
18890                                  <1>
18891                                  <1>       ; Save current drive and reset mode
```

```
18892                                  <1>       ; for 'reset_working_path' procedure (for MainProg)
18893 000120A6 30C0                    <1>       xor    al, al
18894 000120A8 66A3[608D0100]          <1>       mov    [SWP_Mode], ax ; ah = 0
18895 000120AE A0[6E810100]            <1>       mov    al, [Current_Drv]
18896 000120B3 FEC4                    <1>       inc    ah ; mov ah, 1
18897 000120B5 66A3[628D0100]          <1>       mov    [SWP_DRV], ax
18898                                  <1>
18899 000120BB 88DA                    <1>       mov    dl, bl
18900 000120BD E8F05AFFFF              <1>       call   change_current_drive
18901 000120C2 730F                    <1>       jnc    short sysdrive_ok
18902                                  <1> sysdrive_err:
18903 000120C4 C705[64030300]0F00-     <1>       mov    dword [u.r0], ERR_DRV_NOT_RDY ; 'drive not ready !'
18903 000120CC 0000                    <1>
18904 000120CE E9DDB2FFFF              <1>       jmp    error
18905                                  <1> sysdrive_ok:
18906 000120D3 A0[6E810100]            <1>       mov    al, [Current_Drv]
18907 000120D8 8A25[54390100]          <1>       mov    ah, [Last_DOS_DiskNo]
18908 000120DE A3[64030300]            <1>       mov    [u.r0], eax
18909 000120E3 E9E8B2FFFF              <1>       jmp    sysret
18910                                  <1>
18911                                  <1>
18912                                  <1> sysdir: ; Get Current (Working) Drive & Directory (for user)
18913                                  <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
18914                                  <1>       ;
18915                                  <1>       ; INPUT ->
18916                                  <1>       ;         EBX = Current directory name buffer address
18917                                  <1>       ;            (Buffer length = 92 bytes)
18918                                  <1>       ; OUTPUT ->
18919                                  <1>       ;         AL = Current drive (0=A: .. 2=C:)
18920                                  <1>       ;        If CF = 1 -> AL = error code
18921                                  <1>       ;
18922                                  <1>       ; Modified Registers: EAX (at the return of system call)
18923                                  <1>       ;
18924                                  <1>       ; Note: Required directory name buffer length may be
18925                                  <1>       ;     <= 92 bytes for current TRDOS 386 version.
18926                                  <1>       ;     (7*12 name chars + 7 slash + 0)
18927                                  <1>
18928 000120E8 89E5                    <1>       mov    ebp, esp
18929 000120EA 83EC60                  <1>       sub    esp, 96
18930 000120ED 53                      <1>       push   ebx ; User's buffer address
18931 000120EE 30D2                    <1>       xor    dl, dl ; 0 = current drive
18932 000120F0 E8B989FFFF              <1>       call   get_current_directory
18933 000120F5 72CD                    <1>       jc     short sysdrive_err ; 'drive not ready !' error
18934 000120F7 89E6                    <1>       mov    esi, esp ; System's buffer address
18935 000120F9 5F                      <1>       pop    edi  ; User's buffer address
18936                                  <1>       ; ecx = transfer (byte) count (<=92)
18937 000120FA E843F4FFFF              <1>       call   transfer_to_user_buffer
18938 000120FF 89EC                    <1>       mov    esp, ebp
18939 00012101 730F                    <1>       jnc    short sysdir_ok
18940                                  <1> sysdir_err:
18941 00012103 C705[64030300]2E00-     <1>       mov    dword [u.r0], ERR_BUFFER  ; 'buffer error !'
18941 0001210B 0000                    <1>
18942 0001210D E99EB2FFFF              <1>       jmp    error
18943                                  <1> sysdir_ok:
18944 00012112 8A0D[6E810100]          <1>       mov    cl, [Current_Drv]
18945 00012118 890D[64030300]          <1>       mov    [u.r0], ecx
18946 0001211E E9ADB2FFFF              <1>       jmp    sysret
18947                                  <1>
18948                                  <1>
18949                                  <1> sysldrvt: ; Get copy of Logical DOS Drive Description Table
18950                                  <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
18951                                  <1>       ;
18952                                  <1>       ; INPUT ->
18953                                  <1>       ;         BL = Logical DOS drive number (zero based)
18954                                  <1>       ;        ECX = Logical DOS drv desc table buffer addr
18955                                  <1>       ;           (Buffer length = 256 bytes)
18956                                  <1>       ; OUTPUT ->
18957                                  <1>       ;          cf = 0 ->
18958                                  <1>       ;               AL = Current Drive number
18959                                  <1>       ;               AH = The Last Logical DOS Drive no.
18960                                  <1>       ;          cf = 1 -> Error code in AL
18961                                  <1>       ;               AH = The Last Logical DOS Drive no.
18962                                  <1>       ;
18963                                  <1>       ; Modified Registers: EAX (at the return of system call)
18964                                  <1>       ;
18965                                  <1>       ; Note: Required description table buffer length is
18966                                  <1>       ;     256 bytes for current TRDOS 386 version.
18967                                  <1>
18968 00012123 89CF                    <1>       mov    edi, ecx ; Destination address (user space)
18969 00012125 88DC                    <1>       mov    ah, bl
18970 00012127 30C0                    <1>       xor    al, al
18971 00012129 BE00010900              <1>       mov    esi, Logical_DOSDisks
18972 0001212E 01C6                    <1>       add    esi, eax ; Source address (system space)
18973 00012130 B900010000              <1>       mov    ecx, 256 ; Byte count
18974                                  <1>                     ; Logical Dos Drv Desc Table size
18975 00012135 E808F4FFFF              <1>       call   transfer_to_user_buffer
18976 0001213A 72C7                    <1>       jc     short sysdir_err
18977 0001213C 8A2D[54390100]          <1>       mov    ch, [Last_DOS_DiskNo]
18978 00012142 EBCE                    <1>       jmp    short sysdir_ok
18979                                  <1>
18980                                  <1>
18981                                  <1> systime: ; Get System Date&Time
18982                                  <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
18983                                  <1>       ;
18984                                  <1>         ; INPUT -> BL =
18985                                  <1>       ;          0 = Get Date&Time in Unix/Epoch format
18986                                  <1>       ;          1 = Get Time in MSDOS format
18987                                  <1>       ;          2 = Get Date in MSDOS format
18988                                  <1>       ;          3 = Get Date&Time in MSDOS format
18989                                  <1>       ;          4 & other values =
18990                                  <1>       ;             System timer ticks will be returned
18991                                  <1>       ;             in EAX and Carry Flag will be set.
18992                                  <1>       ;             (CF will not be set if BL = 4)
18993                                  <1>       ; OUTPUT ->
18994                                  <1>       ;     For BL input = 3
```

```
18995                                  <1>       ;          EAX = Current Time (RTC)
18996                                  <1>       ;            AL = Second (DL in MSDOS)
18997                                  <1>       ;            AH = Minute (CL in MSDOS)
18998                                  <1>       ;            HW of EAX = Hour (CH in MSDOS)
18999                                  <1>       ;         EDX = Current System Date (RTC)
19000                                  <1>       ;            DL = Day (DL in MSDOS)
19001                                  <1>       ;            DH = Month (DH in MSDOS)
19002                                  <1>       ;            HW of EDX = Year (CX in MSDOS)
19003                                  <1>       ;
19004                                  <1>       ;      For BL input = 2
19005                                  <1>       ;         EAX = Current System Date (RTC)
19006                                  <1>       ;            DL = Day (DL in MSDOS)
19007                                  <1>       ;            DH = Month (DH in MSDOS)
19008                                  <1>       ;            HW of EDX = Year (CX in MSDOS)
19009                                  <1>       ;
19010                                  <1>       ;      For BL input = 1
19011                                  <1>       ;          EAX = Current Time (RTC)
19012                                  <1>       ;            AL = Second (DL in MSDOS)
19013                                  <1>       ;            AH = Minute (CL in MSDOS)
19014                                  <1>       ;            HW of EAX = Hour (CH in MSDOS)
19015                                  <1>       ;
19016                                  <1>       ;      For BL input = 0
19017                                  <1>       ;          EAX = Unix (Epoch) Time Ticks/Seconds
19018                                  <1>       ;
19019                                  <1>       ;      For BL input  = 4
19020                                  <1>       ;         EAX = System timer ticks
19021                                  <1>       ;
19022                                  <1>       ;      If CF = 1 (for other values of BL input)
19023                                  <1>       ;         EAX = System timer ticks (no error code!)
19024                                  <1>       ;
19025                                  <1>       ; Modified Registers: EAX, (EDX)
19026                                  <1>       ;              (at the return of system call)
19027                                  <1>       ;
19028                                  <1>
19029 00012144 20DB                   <1>       and   bl, bl
19030 00012146 750F                   <1>       jnz   short systime_1
19031 00012148 E85950FFFF             <1>       call  epoch
19032                                  <1> systime_0:
19033 0001214D A3[64030300]           <1>       mov   [u.r0], eax
19034 00012152 E979B2FFFF             <1>       jmp   sysret
19035                                  <1> systime_1:
19036 00012157 80FB04                 <1>       cmp   bl, 4
19037 0001215A 7211                   <1>       jb    short systime_2
19038 0001215C A1[28810100]           <1>       mov   eax, [TIMER_LH] ; 18.2 Hz timer ticks
19039                                  <1>                             ; Note: [TIMER_LH] may be set
19040                                  <1>                             ; to wrong timer value due to
19041                                  <1>                             ; program functions.
19042                                  <1>                             ; (This value must not be
19043                                  <1>                             ; accepted as [TIMER_LH]/18.2
19044                                  <1>                             ; seconds since the midnight.)
19045 00012161 76EA                   <1>       jna   short systime_0
19046 00012163 A3[64030300]           <1>       mov   [u.r0], eax
19047 00012168 E943B2FFFF             <1>       jmp   error ; cf = 1 & [u.r0] = eax = timer ticks
19048                                  <1>
19049                                  <1> systime_2:
19050                                  <1>       ;push  ebx
19051 0001216D E8AE4FFFFF             <1>       call  get_rtc_date_time
19052                                  <1>       ;pop   ebx
19053 00012172 F6C301                 <1>       test  bl, 1
19054 00012175 7429                   <1>       jz    short systime_4
19055 00012177 30E4                   <1>       xor   ah, ah
19056 00012179 A0[767D0100]           <1>       mov   al, [hour]
19057 0001217E 88C2                   <1>       mov   dl, al
19058 00012180 C1E010                 <1>       shl   eax, 16
19059 00012183 A0[7A7D0100]           <1>       mov   al, [second]
19060 00012188 8A25[787D0100]         <1>       mov   ah, [minute]
19061 0001218E F6C302                 <1>       test  bl, 2
19062 00012191 74BA                   <1>       jz    short systime_0
19063                                  <1>       ; Check time & date match risk
19064                                  <1>       ; (23:59:59 may cause to wrong
19065                                  <1>       ; date -new day with previous date-...)
19066 00012193 80FA17                 <1>       cmp   dl, 23
19067 00012196 7206                   <1>       jb    short systime_3
19068 00012198 663D3B3B               <1>       cmp   ax, (59*256)+59 ; if hour is 23:59:59
19069 0001219C 73CF                   <1>       jnb   short systime_2 ; wait for 1 second
19070                                  <1> systime_3:
19071                                  <1>       ; eax = time
19072 0001219E 89C6                   <1>       mov   esi, eax
19073                                  <1> systime_4:
19074 000121A0 66A1[707D0100]         <1>       mov   ax, [year]
19075 000121A6 C1E010                 <1>       shl   eax, 16
19076 000121A9 A0[747D0100]           <1>       mov   al, [day]
19077 000121AE 8A25[727D0100]         <1>       mov   ah, [month]
19078                                  <1>       ; eax = date
19079 000121B4 80E301                 <1>       and   bl, 1
19080 000121B7 7494                   <1>       jz    short systime_0
19081 000121B9 96                     <1>       xchg  esi, eax
19082                                  <1>       ; eax = time, esi = date
19083 000121BA 8B2D[60030300]         <1>       mov   ebp, [u.usp]  ; EBP points to user's registers
19084                                  <1>       ; (user) edx <-- (system) esi
19085 000121C0 897514                 <1>       mov   [ebp+20], esi ; return to user with EDX value
19086 000121C3 EB88                   <1>       jmp   short systime_0
19087                                  <1>
19088                                  <1>
19089                                  <1> sysstime: ; Set System Date&Time
19090                                  <1>       ; 31/12/2017
19091                                  <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
19092                                  <1>       ;
19093                                  <1>         ; INPUT -> BL =
19094                                  <1>       ;         0 = Set Date&Time in Unix/Epoch format
19095                                  <1>       ;         1 = Set Time in MSDOS format
19096                                  <1>       ;         2 = Set Date in MSDOS format
19097                                  <1>       ;         3 = Set Date&Time in MSDOS format
19098                                  <1>       ;         4 = Set System Timer (Ticks)
19099                                  <1>       ;         5 = Convert/Save current time to/as
```

```
19100                                  <1>       ;            18.2 Hz system timer ticks
19101                                  <1>       ;          6 = Convert MSDOS Date&Time to UNIX format
19102                                  <1>       ;             without setting system date&time ; (test)
19103                                  <1>       ;          7 = Convert UNIX Date&Time to MSDOS format
19104                                  <1>       ;             without setting system date&time ; (test)
19105                                  <1>       ;         8-0FFh = invalid !
19106                                  <1>       ;        ECX = Time (or Timer) value in selected format
19107                                  <1>       ;        EDX = Date value in MSDOS format if BL=2,3,6
19108                                  <1>       ;
19109                                  <1>       ; OUTPUT ->
19110                                  <1>       ;      If CF = 0 ->
19111                                  <1>       ;           EAX = Set value
19112                                  <1>       ;      If CF = 1 -> (invalid BL input)
19113                                  <1>       ;         EAX = Ticks count [TIMER_LH]
19114                                  <1>       ;
19115                                  <1>
19116 000121C5 20DB                   <1>       and    bl, bl ; 0
19117 000121C7 7511                   <1>       jnz    short sysstime_0
19118 000121C9 89C8                   <1>       mov    eax, ecx
19119 000121CB E86050FFFF             <1>       call   convert_from_epoch
19120 000121D0 E80851FFFF             <1>       call   set_rtc_date_time
19121 000121D5 E9F6B1FFFF             <1>       jmp    sysret
19122                                  <1> sysstime_0:
19123 000121DA 80FB08                 <1>       cmp    bl, 8
19124 000121DD 722D                   <1>       jb     short sysstime_1
19125                                  <1>       ; invalid input (>7)
19126 000121DF A1[28810100]           <1>       mov    eax, [TIMER_LH] ; 18.2 Hz timer ticks
19127                                  <1>                               ; Note: [TIMER_LH] may be set
19128                                  <1>                               ; to wrong timer value due to
19129                                  <1>                               ; program functions.
19130                                  <1>                               ; (This value must not be
19131                                  <1>                               ; accepted as [TIMER_LH]/18.2
19132                                  <1>                               ; seconds since the midnight.)
19133 000121E4 A3[64030300]           <1>       mov    [u.r0], eax
19134 000121E9 E9C2B1FFFF             <1>       jmp    error ; cf = 1 & [u.r0] = eax = timer ticks
19135                                  <1>
19136                                  <1> sysstime_8:
19137                                  <1>       ; BL = 7
19138 000121EE 89C8                   <1>       mov    eax, ecx ; seconds since 1/1/1970 00:00:00
19139 000121F0 E83B50FFFF             <1>       call   convert_from_epoch
19140 000121F5 30E4                   <1>       xor    ah, ah
19141 000121F7 A0[767D0100]           <1>       mov    al, [hour]
19142 000121FC C1E010                 <1>       shl    eax, 16
19143 000121FF A0[7A7D0100]           <1>       mov    al, [second]
19144 00012204 8A25[787D0100]         <1>       mov    ah, [minute]
19145 0001220A EB92                   <1>       jmp    short systime_3
19146                                  <1>
19147                                  <1> sysstime_1:
19148 0001220C 80FB04                 <1>       cmp    bl, 4
19149 0001220F 743F                   <1>       je     short sysstime_2 ; set system timer ticks
19150 00012211 80FB05                 <1>       cmp    bl, 5
19151 00012214 754B                   <1>       jne    short sysstime_4
19152                                  <1>       ; convert current time to system timer ticks (18.2Hz)
19153 00012216 E8054FFFFF             <1>       call   get_rtc_date_time
19154 0001221B 0FB60D[767D0100]       <1>       movzx  ecx, byte [hour]
19155 00012222 B8100E0000             <1>       mov    eax, 60*60 ; 1 hour = 3600 seconds
19156 00012227 F7E1                   <1>       mul    ecx
19157 00012229 89C3                   <1>       mov    ebx, eax
19158 0001222B B13C                   <1>       mov    cl, 60  ; 1 minute = 60 seconds
19159 0001222D 0FB605[787D0100]       <1>       movzx  eax, byte [minute]
19160 00012234 F7E1                   <1>       mul    ecx
19161 00012236 01D8                   <1>       add    eax, ebx
19162 00012238 8A0D[7A7D0100]         <1>       mov    cl, [second]
19163 0001223E 01C8                   <1>       add    eax, ecx
19164 00012240 B1B6                   <1>       mov    cl, 182
19165 00012242 F7E1                   <1>       mul    ecx
19166 00012244 83C009                 <1>       add    eax, 9
19167 00012247 83D200                 <1>       adc    edx, 0
19168 0001224A B10A                   <1>       mov    cl, 10
19169 0001224C F7F1                   <1>       div    ecx
19170                                  <1>       ; eax = ((182*seconds)+9)/10
19171 0001224E 89C1                   <1>       mov    ecx, eax
19172                                  <1> sysstime_2:
19173 00012250 890D[28810100]         <1>       mov    [TIMER_LH], ecx ; 18.2 * seconds
19174                                  <1> sysstime_3:
19175 00012256 890D[64030300]         <1>       mov    [u.r0], ecx
19176 0001225C E96FB1FFFF             <1>       jmp    sysret
19177                                  <1> sysstime_4:
19178 00012261 80FB06                 <1>       cmp    bl, 6
19179 00012264 7788                   <1>       ja     short sysstime_8
19180                                  <1>
19181 00012266 890D[64030300]         <1>       mov    [u.r0], ecx
19182                                  <1>
19183 0001226C 880D[7A7D0100]         <1>       mov    [second], cl
19184 00012272 882D[787D0100]         <1>       mov    [minute], ch
19185 00012278 C1E910                 <1>       shr    ecx, 16
19186 0001227B 880D[767D0100]         <1>       mov    [hour], cl
19187                                  <1>       ; BL = 1,2,3,6
19188 00012281 80FB01                 <1>       cmp    bl, 1
19189 00012284 762A                   <1>       jna    short sysstime_5
19190                                  <1>       ; BL = 2,3,6
19191 00012286 8815[747D0100]         <1>       mov    [day], dl
19192 0001228C 8835[727D0100]         <1>       mov    [month], dh
19193 00012292 C1EA10                 <1>       shr    edx, 16
19194 00012295 668915[707D0100]       <1>       mov    [year], dx
19195 0001229C 80E303                 <1>       and    bl, 3
19196 0001229F 742D                   <1>       jz     short sysstime_7 ; 6
19197                                  <1>       ; BL = 2,3
19198 000122A1 F6C301                 <1>       test   bl, 1
19199 000122A4 7419                   <1>       jz     short sysstime_6 ; 2
19200                                  <1>       ; BL = 3
19201 000122A6 E83250FFFF             <1>       call   set_rtc_date_time
19202 000122AB E920B1FFFF             <1>       jmp    sysret
19203                                  <1> sysstime_5:
19204                                  <1>       ; BL = 1
```

```
19205 000122B0 E86950FFFF          <1>       call   set_time_bcd
19206 000122B5 E81C43FFFF          <1>       call   set_rtc_time
19207 000122BA E911B1FFFF          <1>       jmp    sysret
19208                              <1> sysstime_6:
19209                              <1>       ; BL = 2
19210 000122BF E82D50FFFF          <1>       call   set_date_bcd
19211 000122C4 E87C43FFFF          <1>       call   set_rtc_date
19212 000122C9 E902B1FFFF          <1>       jmp    sysret
19213                              <1> sysstime_7:
19214                              <1>       ; BL = 6
19215                              <1>       ; [year], [month], [day],
19216                              <1>       ; [hour], [minute], [second]
19217 000122CE E8D84EFFFF          <1>       call   convert_to_epoch
19218 000122D3 89C1                <1>       mov    ecx, eax ; seconds since 1/1/1970 00:00:00
19219 000122D5 E97CFFFFFF          <1>       jmp    sysstime_3
19220                              <1>
19221                              <1> sysrename: ; Rename File (or Directory)
19222                              <1>       ; 19/01/2021
19223                              <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
19224                              <1>       ;
19225                              <1>         ; INPUT ->
19226                              <1>         ;          EBX = File/Directory (ASCIIZ) name address
19227                              <1>       ;         ECX = New name (in same dir, no path name)
19228                              <1>       ; OUTPUT ->
19229                              <1>       ;          cf = 0 -> EAX = 0
19230                              <1>       ;          cf = 1 -> Error code in AL
19231                              <1>
19232                              <1>       ; 19/01/2021
19233 000122DA 803D[B0030300]00    <1>       cmp    byte [u.uid], 0  ; root (super user) ?
19234 000122E1 7614                <1>       jna    short sysrename_0
19235                              <1>
19236                              <1> sysrename_perm_err:
19237                              <1>       ;mov   dword [u.r0], ERR_PERM_DENIED
19238 000122E3 B80B000000          <1>       mov    eax, ERR_PERM_DENIED ; 'permission denied !'
19239 000122E8 A3[64030300]        <1>       mov    [u.r0], eax
19240 000122ED A3[C8030300]        <1>       mov    [u.error], eax
19241 000122F2 E9B9B0FFFF          <1>       jmp    error
19242                              <1>
19243                              <1> sysrename_0:
19244 000122F7 51                  <1>       push   ecx ; new file name address (in user space)
19245 000122F8 89DE                <1>       mov    esi, ebx
19246                              <1>       ; file name is forced, change directory as temporary
19247                              <1>       ;mov   ax, 1
19248                              <1>       ;mov   [FFF_Valid], ah ; 0 ; reset
19249                              <1>       ;call  set_working_path
19250 000122FA E8BF060000          <1>       call   set_working_path_x
19251 000122FF 731E                <1>       jnc    short sysrename_1
19252 00012301 21C0                <1>       and    eax, eax  ; 0 -> Bad Path!
19253 00012303 7505                <1>       jnz    short sysrename_err
19254                              <1>       ; eax = 0
19255                              <1> sysrename_path_not_found:
19256 00012305 B813000000          <1>       mov    eax, ERR_INV_PATH_NAME ; 'Bad path name !'
19257                              <1> sysrename_err:
19258 0001230A 59                  <1>       pop    ecx ; new file name address (in user space)
19259                              <1> sysrename_error:
19260 0001230B A3[64030300]        <1>       mov    [u.r0], eax
19261 00012310 A3[C8030300]        <1>       mov    [u.error], eax
19262 00012315 E879070000          <1>       call   reset_working_path
19263 0001231A E991B0FFFF          <1>       jmp    error
19264                              <1> sysrename_1:
19265 0001231F B008                <1>       mov    al, 08h ; Except volume labels (& long names)
19266 00012321 A0[248B0100]        <1>       mov    al, [Attributes]
19267 00012326 2410                <1>       and    al, 10h ;
19268                              <1>       ;mov   esi, FindFile_Name
19269                              <1>       ;mov   ax, 1800h ; Only files
19270                              <1>       ;mov   ax, 0810h ; Only directories
19271 00012328 66B80008            <1>       mov    ax, 0800h ; Find File or Directory
19272 0001232C E8246CFFFF          <1>       call   find_first_file
19273                              <1>       ;jnc   short sysrename_2
19274 00012331 72D7                <1>       jc     short sysrename_err
19275                              <1> sysrename_2:
19276                              <1>       ; ESI = Directory Entry (FindFile_DirEntry) Location
19277                              <1>       ; EDI = Directory Buffer Directory Entry Location
19278                              <1>       ; EAX = File Size
19279                              <1>       ;  BL = Attributes of The File/Directory
19280                              <1>       ;  BH = Long Name Yes/No Status (>0 is YES)
19281                              <1>       ;  DX > 0 : Ambiguous filename chars are used
19282                              <1>
19283 00012333 6621D2              <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
19284 00012336 7407                <1>       jz     short sysrename_3
19285 00012338 B81A000000          <1>       mov    eax, ERR_INV_FILE_NAME ; 'invalid file name !'
19286 0001233D EBCB                <1>         jmp  short sysrename_err
19287                              <1> sysrename_3:
19288                              <1>       ; EDI = Directory buffer entry offset/address
19289                              <1>       ; BL = File (or Directory) Attributes
19290                              <1>       ; mov  bl, [EDI+0Bh]
19291                              <1>
19292 0001233F 5A                  <1>       pop    edx ; new file name address (in user space)
19293                              <1>
19294                              <1>       ; check file/directory attributes
19295 00012340 F6C307              <1>       test   bl, 7 ; system, hidden, readonly
19296 00012343 759E                <1>         jnz  short sysrename_perm_err
19297                              <1> sysrename_4:
19298 00012345 66817F0CA101        <1>       cmp    word [edi+DirEntry_NTRes], 01A1h ; Singlix FS
19299 0001234B 7496                <1>       je     short sysrename_perm_err ; -temporary!-
19300                              <1>
19301                              <1>       ; save old file name & file info (FFF structure)
19302 0001234D BE[0E8A0100]        <1>       mov    esi, FindFile_Drv
19303 00012352 BF[548B0100]        <1>       mov    edi, SourceFile_Drv
19304 00012357 B920000000          <1>       mov    ecx, 128/4
19305 0001235C F3A5                <1>       rep    movsd
19306                              <1>
19307 0001235E 89D6                <1>       mov    esi, edx ; new file name address (in user space)
19308 00012360 BF[D48B0100]        <1>       mov    edi, DestinationFile_Drv
19309 00012365 E8BC8DFFFF          <1>       call   parse_path_name
```

```
19310 0001236A 729F               <1>       jc     short sysrename_error ; eax = 1 (Bad file name)
19311                             <1>
19312                             <1>       ; same drive ?
19313 0001236C A0[0E8A0100]       <1>       mov    al, [FindFile_Drv]
19314 00012371 3A05[D48B0100]     <1>       cmp    al, [DestinationFile_Drv]
19315                             <1>       ;jne   short sysrename_perm_err ; Permission denied
19316 00012377 7509               <1>       jne    short sysrename_5 ; Bad file name
19317                             <1>
19318                             <1>       ; no path name !? (rename file in same directory)
19319 00012379 803D[D58B0100]20   <1>       cmp    byte [DestinationFile_Directory], 20h
19320 00012380 7607               <1>       jna    short sysrename_6
19321                             <1> sysrename_5:
19322 00012382 B801000000         <1>       mov    eax, ERR_BAD_CMD_ARG ; 1 = Bad file name
19323                             <1>                                   ; (Bad argument)
19324 00012387 EB82               <1>       jmp    short sysrename_error
19325                             <1> sysrename_6:
19326 00012389 803D[168C0100]20   <1>       cmp    byte [DestinationFile_Name], 20h
19327 00012390 76F0               <1>       jna    short sysrename_5
19328                             <1>
19329 00012392 BE[168C0100]       <1>       mov    esi, DestinationFile_Name
19330 00012397 E8776FFFFF         <1>       call   check_filename ; is it a valid msdos file name?
19331 0001239C 0F8269FFFFFF       <1>       jc     sysrename_error ; 26 = ERR_INV_FILE_NAME
19332                             <1>
19333                             <1>       ;mov   esi, DestinationFile_Name
19334 000123A2 66B80008           <1>       mov    ax, 0800h ; Find File or Directory
19335 000123A6 E8AA6BFFFF         <1>       call   find_first_file
19336 000123AB 720A               <1>       jc     short sysrename_7
19337                             <1>
19338 000123AD B80E000000         <1>       mov    eax, ERR_FILE_EXISTS  ; file already exists !
19339 000123B2 E954FFFFFF         <1>       jmp    sysrename_error
19340                             <1> sysrename_7:
19341                             <1>       ; eax = 2 (File not found !)
19342 000123B7 3C02               <1>       cmp    al, 2 ; ERR_NOT_FOUND
19343 000123B9 0F854CFFFFFF       <1>       jne    sysrename_error
19344                             <1>
19345                             <1>       ; 31/12/2017
19346                             <1>       ; Following code is also part of 'rename_file' in
19347                             <1>       ; 'trdosk3.s' (MainProg's 'rename' command) ; 13/11/2017
19348 000123BF BE[168C0100]       <1>       mov    esi, DestinationFile_Name ; (Rename_NewName)
19349 000123C4 668B0D[CE8B0100]   <1>       mov    cx, [SourceFile_DirEntryNumber]
19350 000123CB 66A1[BA8B0100]     <1>       mov    ax, [SourceFile_DirEntry+20] ; First Cluster, HW
19351 000123D1 C1E010             <1>       shl    eax, 16
19352 000123D4 66A1[C08B0100]     <1>       mov    ax, [SourceFile_DirEntry+26] ; First Cluster, LW
19353 000123DA 0FB61D[A38B0100]   <1>       movzx  ebx, byte [SourceFile_LongNameEntryLength]
19354 000123E1 E88895FFFF         <1>       call   rename_directory_entry
19355 000123E6 0F821FFFFFFF       <1>       jc     sysrename_error
19356                             <1>       ;xor   eax, eax
19357 000123EC A3[64030300]       <1>       mov    [u.r0], eax ; 0
19358                             <1>       ;mov   [u.error], eax
19359 000123F1 E89D060000         <1>       call   reset_working_path
19360 000123F6 E9D5AFFFFF         <1>       jmp    sysret
19361                             <1>
19362                             <1> sysmem: ; Get Total&Free Memory amount
19363                             <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
19364                             <1>       ;
19365                             <1>         ; INPUT ->
19366                             <1>       ;     none
19367                             <1>       ; OUTPUT ->
19368                             <1>       ;     EAX = Total memory count (in bytes)
19369                             <1>       ;     EBX = Virtually available memory amount (in bytes)
19370                             <1>       ;           = 4GB - CORE (4MB)
19371                             <1>       ;     ECX = Free memory count (in bytes)
19372                             <1>       ;     EDX = Calculated free memory count (in bytes)
19373                             <1>
19374 000123FB A1[AC800100]       <1>       mov    eax, [memory_size] ; in pages
19375 00012400 C1E00C             <1>       shl    eax, 12               ; in bytes
19376 00012403 A3[64030300]       <1>       mov    [u.r0], eax
19377 00012408 E87E1EFFFF         <1>       call   calc_free_mem
19378                             <1>       ; edx = calculated free pages
19379                             <1>       ; ecx = 0
19380 0001240D 8B2D[60030300]     <1>       mov    ebp, [u.usp]  ; EBP points to user's registers
19381 00012413 C745100000C0FF     <1>       mov    dword [ebp+16], ECORE ; EBX (for user)
19382                             <1>                              ; 0FFC00000h ; 4GB - 4MB
19383 0001241A C1E20C             <1>       shl    edx, 12
19384 0001241D 895514             <1>       mov    [ebp+20], edx ; EDX (for user)
19385 00012420 8B0D[B0800100]     <1>       mov    ecx, [free_pages]
19386 00012426 C1E10C             <1>       shl    ecx, 12       ; free bytes
19387 00012429 894D18             <1>       mov    [ebp+24], ecx ; ECX (for user)
19388                             <1>       ;mov   [free_pages], edx
19389 0001242C E99FAFFFFF         <1>       jmp    sysret
19390                             <1>
19391                             <1> sysprompt:
19392                             <1>       ; Set TRDOS 386 Command Interpreter (MainProg) prompt
19393                             <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
19394                             <1>       ;
19395                             <1>       ; INPUT ->
19396                             <1>       ;     EBX = 0 -> use default prompt
19397                             <1>       ;     EBX > 0 -> prompt string (ASCIIZ) address
19398                             <1>       ;                (Max. 11 characters except ZERO tail)
19399                             <1>       ; OUTPUT ->
19400                             <1>       ;     (EAX = 0)
19401                             <1>       ;     CF = 0 -> Successful
19402                             <1>       ;     CF = 1 -> Failed
19403                             <1>
19404 00012431 21DB               <1>       and    ebx, ebx
19405 00012433 750A               <1>       jnz    short sysprompt_0
19406                             <1>
19407 00012435 E81F65FFFF         <1>       call   default_command_prompt ; '['+'TRDOS'+']'
19408 0001243A E991AFFFFF         <1>       jmp    sysret
19409                             <1>
19410                             <1> sysprompt_0:
19411 0001243F 31C0               <1>       xor    eax, eax
19412 00012441 A3[64030300]       <1>       mov    [u.r0], eax
19413 00012446 89DE               <1>       mov    esi, ebx
19414 00012448 B90C000000         <1>       mov    ecx, 12
```

```
19415 0001244D 89E5              <1>       mov    ebp, esp
19416 0001244F 29CC              <1>       sub    esp, ecx
19417 00012451 49                <1>       dec    ecx ; 11
19418 00012452 89E7              <1>       mov    edi, esp
19419 00012454 E833F1FFFF        <1>       call   transfer_from_user_buffer
19420 00012459 7211              <1>       jc     short sysprompt_err
19421 0001245B 803E20            <1>       cmp    byte [esi], 20h
19422 0001245E 760C              <1>       jna    short sysprompt_err
19423 00012460 E80665FFFF        <1>       call   set_command_prompt
19424 00012465 89EC              <1>       mov    esp, ebp
19425 00012467 E964AFFFFF        <1>       jmp    sysret
19426                            <1> sysprompt_err:
19427                            <1> syspath_err:
19428 0001246C 89EC              <1>       mov    esp, ebp
19429 0001246E E93DAFFFFF        <1>       jmp    error
19430                            <1>
19431                            <1> syspath:
19432                            <1>       ; Get/Set Run Path
19433                            <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
19434                            <1>       ;
19435                            <1>         ; INPUT ->
19436                            <1>       ;      EBX = 0 -> get path (to buffer address in ECX)
19437                            <1>       ;      EBX > 0 -> set path
19438                            <1>       ;          EBX = Path string buffer address (ASCIIZ)
19439                            <1>       ;               (Path description except 'PATH=')
19440                            <1>       ;      ECX = Buffer address (if EBX = 0)
19441                            <1>       ;            (ECX will not be used if EBX > 0)
19442                            <1>       ;      DL = Buffer size (0 = 256 byte)
19443                            <1>       ;
19444                            <1>       ; OUTPUT ->
19445                            <1>       ;      CF = 0 -> Successful (EAX = String length)
19446                            <1>       ;      CF = 1 -> Failed (EAX = 0)
19447                            <1>       ;
19448                            <1>       ; NOTE: 'PATH=' or 'PATH' must be excluded
19449                            <1>       ;  (It must not be at the beginning of the string.)
19450                            <1>
19451 00012473 89E5              <1>       mov    ebp, esp
19452 00012475 81EC00010000      <1>       sub    esp, 256
19453 0001247B 89E7              <1>       mov    edi, esp
19454                            <1>
19455 0001247D 31C0              <1>       xor    eax, eax
19456 0001247F A3[64030300]      <1>       mov    [u.r0], eax
19457                            <1>
19458 00012484 21DB              <1>       and    ebx, ebx
19459 00012486 752E              <1>       jnz    short syspath_0
19460                            <1>
19461                            <1>       ; EBX = 0 -> get run path
19462 00012488 89CB              <1>       mov    ebx, ecx ; buffer addr (in user's mem space)
19463 0001248A BE[213A0100]      <1>       mov    esi, Cmd_Path  ; 'PATH' address
19464 0001248F 0FB6CA            <1>       movzx  ecx, dl
19465 00012492 80E901            <1>       sub    cl, 1 ; 0 -> 255, 1 -> 0
19466 00012495 6683D101          <1>       adc    cx, 1 ; 255 -> 256, 0 -> 1
19467                            <1>       ; EDI = Output buffer
19468                            <1>       ; CX = Buffer length
19469                            <1>       ; AL = 0 -> use ASCIIZ word in [ESI]
19470                            <1>       ; ESI = 'PATH' address (with zero tail)
19471 00012499 E8017DFFFF        <1>       call   get_environment_string
19472 0001249E 72CC              <1>       jc     short syspath_err
19473 000124A0 89DF              <1>       mov    edi, ebx ; User's buffer address
19474 000124A2 89E6              <1>       mov    esi, esp
19475                            <1>       ; EDI = User's buffer address
19476                            <1>       ; ECX = transfer (byte) count
19477 000124A4 E899F0FFFF        <1>       call   transfer_to_user_buffer
19478 000124A9 72C1              <1>       jc     short syspath_err
19479 000124AB 890D[64030300]    <1>       mov    [u.r0], ecx
19480 000124B1 E91AAFFFFF        <1>       jmp    sysret
19481                            <1>
19482                            <1> syspath_0:
19483 000124B6 89DE              <1>       mov    esi, ebx
19484 000124B8 0FB6CA            <1>       movzx  ecx, dl
19485 000124BB 80E901            <1>       sub    cl, 1 ; 0 -> 255, 1 -> 0
19486 000124BE 6683D101          <1>       adc    cx, 1 ; 255 -> 256, 0 -> 1
19487 000124C2 E8C5F0FFFF        <1>       call   transfer_from_user_buffer
19488 000124C7 72A3              <1>       jc     short syspath_err
19489                            <1>       ;(*) 'PATH=' will be added to
19490                            <1>       ;         the head of the string
19491 000124C9 83EC08            <1>       sub    esp, 8 ;(*)
19492 000124CC 89FE              <1>       mov    esi, edi ;(*)
19493 000124CE E8B07CFFFF        <1>       call   set_path_x ;(*)
19494 000124D3 7297              <1>       jc     short syspath_err
19495 000124D5 8915[64030300]    <1>       mov    [u.r0], edx ; run path string length
19496 000124DB E9F0AEFFFF        <1>       jmp    sysret
19497                            <1>
19498                            <1> sysenv:
19499                            <1>       ; Get/Set Environment Variables
19500                            <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
19501                            <1>       ;
19502                            <1>         ; INPUT ->
19503                            <1>       ;      EBX = 0 -> get (all) environment variables
19504                            <1>       ;            (Required Buffer length = 512 bytes)
19505                            <1>       ;      EBX > 0 -> set (one) environment variable
19506                            <1>       ;            (If there is not a '=' after
19507                            <1>       ;            the environment variable name, it will
19508                            <1>       ;            accepted as 'get environment variable'.)
19509                            <1>       ;             EBX = Buffer address
19510                            <1>       ;      ECX = Buffer address (if EBX = 0)
19511                            <1>       ;            (ECX will not be used if EBX > 0)
19512                            <1>       ;            (Note: Buffer size is 512 bytes.)
19513                            <1>       ;      DL = Buffer size (0 = 256 byte)
19514                            <1>       ;           (For one envrionment variable)
19515                            <1>       ;
19516                            <1>       ; OUTPUT ->
19517                            <1>       ;      (EAX = 0)
19518                            <1>       ;      CF = 0 -> Successful (EAX = String length)
19519                            <1>       ;      CF = 1 -> Failed (EAX = 0)
```

```
19520                           <1>       ;
19521                           <1>       ; Note: Environment variable name, for example,
19522                           <1>       ;     'PATH=' must be included at the beginning
19523                           <1>       ;     of the environment string. If the variable
19524                           <1>       ;     name is as 'PATH' but it is not as 'PATH='
19525                           <1>       ;     the variable string (row) will be returned.
19526                           <1>       ;     If variable name is as 'PATH=' but there is
19527                           <1>       ;     not a following text after the variable name,
19528                           <1>       ;     the environment variable will be reset/deleted.
19529                           <1>
19530 000124E0 89E5             <1>       mov   ebp, esp
19531 000124E2 81EC00020000     <1>       sub   esp, 512
19532 000124E8 89E7             <1>       mov   edi, esp
19533                           <1>
19534 000124EA 31C0             <1>       xor   eax, eax
19535 000124EC A3[64030300]     <1>       mov   [u.r0], eax
19536                           <1>
19537 000124F1 21DB             <1>       and   ebx, ebx
19538 000124F3 7524             <1>       jnz   short sysenv_0
19539                           <1>
19540                           <1>       ; EBX = 0 -> get (all) environment variables
19541 000124F5 89EC             <1>       mov   esp, ebp
19542 000124F7 BE00300900       <1>       mov   esi, Env_Page  ; Environment page
19543 000124FC 89CF             <1>       mov   edi, ecx ; buffer addr (in user's mem space)
19544 000124FE B900020000       <1>       mov   ecx, 512
19545 00012503 E83AF0FFFF       <1>       call  transfer_to_user_buffer
19546 00012508 0F82A2AEFFFF     <1>       jc    error
19547 0001250E 890D[64030300]   <1>       mov   [u.r0], ecx
19548 00012514 E9B7AEFFFF       <1>       jmp   sysret
19549                           <1>
19550                           <1> sysenv_0:
19551 00012519 89DE             <1>       mov   esi, ebx ; * ; user's buffer address
19552 0001251B 0FB6CA           <1>       movzx ecx, dl
19553 0001251E 80E901           <1>       sub   cl, 1 ; 0 -> 255, 1 -> 0
19554 00012521 6683D101         <1>       adc   cx, 1 ; 255 -> 256, 0 -> 1
19555 00012525 E862F0FFFF       <1>       call  transfer_from_user_buffer
19556 0001252A 723F             <1>       jc    short sysenv_err
19557 0001252C 89FE             <1>       mov   esi, edi
19558 0001252E 8A06             <1>       mov   al, [esi]
19559 00012530 3C20             <1>       cmp   al, 20h
19560 00012532 7637             <1>       jna   short sysenv_err
19561 00012534 3C3D             <1>       cmp   al, '='
19562 00012536 7433             <1>       je    short sysenv_err
19563 00012538 56               <1>       push  esi
19564                           <1> sysenv_1:
19565 00012539 46               <1>       inc   esi
19566 0001253A 803E3D           <1>       cmp   byte [esi], '='
19567 0001253D 7433             <1>       je    short sysenv_3
19568 0001253F 803E20           <1>       cmp   byte [esi], 20h
19569 00012542 73F5             <1>       jnb   short sysenv_1
19570 00012544 C60600           <1>       mov   byte [esi], 0
19571 00012547 5E               <1>       pop   esi
19572                           <1>       ; EDI = Output buffer
19573                           <1>       ; CX = Buffer length
19574 00012548 30C0             <1>       xor   al, al
19575                           <1>       ; AL = 0 -> use ASCIIZ word in [ESI]
19576                           <1>       ; ESI = Environment variable name address
19577 0001254A E8507CFFFF       <1>       call  get_environment_string
19578 0001254F 721A             <1>       jc    short sysenv_err
19579 00012551 89DF             <1>       mov   edi, ebx ; * ; user's buffer address
19580 00012553 89C1             <1>       mov   ecx, eax ; String length
19581 00012555 89E6             <1>       mov   esi, esp
19582                           <1>       ; ESI = system buffer address
19583                           <1>       ; EDI = User's buffer address
19584                           <1>       ; ECX = transfer (byte) count
19585 00012557 E8E6EFFFFF       <1>       call  transfer_to_user_buffer
19586 0001255C 720D             <1>       jc    short sysenv_err
19587 0001255E 890D[64030300]   <1>       mov   [u.r0], ecx ; transfer (byte) count
19588                           <1> sysenv_2:
19589 00012564 89EC             <1>       mov   esp, ebp
19590 00012566 E965AEFFFF       <1>       jmp   sysret
19591                           <1> sysenv_err:
19592 0001256B 89EC             <1>       mov   esp, ebp
19593 0001256D E93EAEFFFF       <1>       jmp   error
19594                           <1> sysenv_3:
19595 00012572 46               <1>       inc   esi
19596 00012573 803E20           <1>       cmp   byte [esi], 20h
19597 00012576 73FA             <1>       jnb   short sysenv_3
19598 00012578 C60600           <1>       mov   byte [esi], 0
19599 0001257B 5E               <1>       pop   esi
19600 0001257C E8E17CFFFF       <1>       call  set_environment_string
19601 00012581 72E8             <1>       jc    short sysenv_err
19602 00012583 8915[64030300]   <1>       mov   [u.r0], edx
19603 00012589 EBD9             <1>       jmp   short sysenv_2
19604                           <1>
19605                           <1>
19606                           <1> ; 22/01/2021
19607                           <1> ; temporary - 24/01/2016
19608                           <1>
19609                           <1> iget:
19610                           <1>       ;retn
19611                           <1> isintr:
19612                           <1>       ;retn
19613                           <1> iopen:
19614                           <1>       ;retn
19615                           <1> iclose:
19616                           <1>       ;retn
19617                           <1> sndc:
19618                           <1>       ;retn
19619                           <1> access:
19620                           <1>       ;retn
19621                           <1> sleep:
19622 0001258B C3               <1>       retn
 3112                               %include 'trdosk7.s' ; 24/01/2016
 3113                           <1> ; ****************************************************************************
```

```
3114                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - DISK READ&WRITE : trdosk7.s
3115                            <1> ; ----------------------------------------------------------------------------
3116                            <1> ; Last Update: 25/02/2016
3117                            <1> ; ----------------------------------------------------------------------------
3118                            <1> ; Beginning: 24/01/2016
3119                            <1> ; ----------------------------------------------------------------------------
3120                            <1> ; Assembler: NASM version 2.11 (trdos386.s)
3121                            <1> ; ----------------------------------------------------------------------------
3122                            <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
3123                            <1> ; DISK_IO.ASM (20/07/2011)
3124                            <1> ; ****************************************************************************
3125                            <1> ; DISK_IO.ASM (c) 2009-2011 Erdogan TAN [ 04/07/2009 ] Last Update: 20/07/2011
3126                            <1>
3127                            <1> disk_write:
3128                            <1>       ; 25/02/2016
3129                            <1>       ; 24/02/2016
3130                            <1>       ; 23/02/2016
3131 0001258C 807E0500         <1>       cmp    byte [esi+LD_LBAYes], 0
3132 00012590 777B             <1>       ja     short lba_write
3133                            <1>
3134                            <1> chs_write:
3135                            <1>       ; 25/02/2016
3136                            <1>       ; 23/02/2016
3137 00012592 C605[5D890100]03  <1>       mov    byte [disk_rw_op], 3 ; CHS write
3138 00012599 EB0D              <1>       jmp    short chs_rw
3139                            <1>
3140                            <1> disk_read:
3141                            <1>       ; 25/02/2016
3142                            <1>       ; 24/02/2016
3143                            <1>       ; 23/02/2016
3144                            <1>       ; 17/02/2016
3145                            <1>       ; 14/02/2016
3146                            <1>       ; 31/01/2016 (TRDOS 386 =  TRDOS v2.0)
3147                            <1>       ; 17/10/2010
3148                            <1>       ; 18/04/2010
3149                            <1>       ;
3150                            <1>       ; INPUT -> EAX = Logical Block Address
3151                            <1>       ;         ESI = Logical Dos Disk Table Offset (DRV)
3152                            <1>       ;         ECX = Sector Count
3153                            <1>       ;         EBX = Destination Buffer
3154                            <1>       ; OUTPUT ->
3155                            <1>       ;         cf = 0 or cf = 1
3156                            <1>       ; (Modified registers: EAX, EBX, ECX, EDX)
3157                            <1>
3158 0001259B 807E0500         <1>       cmp    byte [esi+LD_LBAYes], 0
3159 0001259F 7775             <1>       ja     short lba_read
3160                            <1>
3161                            <1> chs_read:
3162                            <1>       ; 25/02/2016
3163                            <1>       ; 24/02/2016
3164                            <1>       ; 23/02/2016
3165                            <1>       ; 31/01/2016 (TRDOS 386 =  TRDOS v2.0)
3166                            <1>       ; 20/07/2011
3167                            <1>       ; 04/07/2009
3168                            <1>       ;
3169                            <1>       ; INPUT -> EAX = Logical Block Address
3170                            <1>       ;         ECX = Number of sectors to read
3171                            <1>       ;         ESI = Logical Dos Disk Table Offset (DRV)
3172                            <1>       ;         EBX = Destination Buffer
3173                            <1>       ; OUTPUT ->
3174                            <1>       ;         cf = 0 or cf = 1
3175                            <1>       ; (Modified registers: EAX; EBX, ECX, EDX)
3176                            <1>
3177                            <1>       ; 23/02/2016
3178 000125A1 C605[5D890100]02  <1>       mov    byte [disk_rw_op], 2 ; CHS read
3179                            <1>
3180                            <1> chs_rw:
3181                            <1>       ;;movzx edx, word [esi+LD_BPB+SecPerTrack]
3182                            <1>       ;movzx edx, byte [esi+LD_BPB+SecPerTrack] ; <= 63
3183                            <1>       ;mov   [disk_rw_spt], dl
3184                            <1>
3185                            <1> chs_read_next_sector:
3186 000125A8 C605[5E890100]04  <1>       mov    byte [retry_count], 4
3187                            <1>
3188                            <1> chs_read_retry:
3189                            <1>       ;mov   [sector_count], ecx ; 23/02/2016
3190                            <1>
3191 000125AF 50                <1>       push   eax                ; Linear sector #
3192 000125B0 51                <1>       push   ecx                ; # of FAT/FILE/DIR sectors
3193                            <1>
3194 000125B1 0FB74E1E          <1>       movzx  ecx, word [esi+LD_BPB+SecPerTrack]
3195                            <1>       ;movzx ecx, byte [disk_rw_spt] ; 23/02/2016
3196 000125B5 29D2              <1>       sub    edx, edx
3197 000125B7 F7F1              <1>       div    ecx
3198                            <1>       ; eax = track, dx (dl ) = sector (on track)
3199                            <1>       ;sub   cl, dl ; 24/02/2016 (spt - sec)
3200                            <1>       ;push  ecx ; *
3201 000125B9 6689D1            <1>       mov    cx, dx             ; Sector (zero based)
3202 000125BC 6641              <1>       inc    cx                 ; To make it 1 based
3203 000125BE 6651              <1>       push   cx
3204 000125C0 668B4E20          <1>       mov    cx, [esi+LD_BPB+Heads]
3205 000125C4 6629D2            <1>       sub    dx, dx
3206 000125C7 F7F1              <1>       div    ecx                ; Convert track to head & cyl
3207                            <1>       ; eax (ax) = cylinder, dx (dl) = head (max. FFh)
3208 000125C9 88D6              <1>       mov    dh, dl
3209 000125CB 6659              <1>       pop    cx                 ; AX=Cyl, DH=Head, CX=Sector
3210 000125CD 8A5602            <1>       mov    dl, [esi+LD_PhyDrvNo]
3211                            <1>
3212 000125D0 88C5              <1>       mov    ch, al             ; NOTE: max. 1023 cylinders !
3213 000125D2 C0CC02            <1>       ror    ah, 2              ; Rotate 2 bits right
3214 000125D5 08E1              <1>       or     cl, ah
3215                            <1>
3216                            <1>       ; 24/02/2016
3217                            <1>       ;pop   eax ; * (spt - sec) (example: 63 - 0 = 63)
3218                            <1>       ;cmp   eax, [sector_count]
```

```
3219                                <1>       ;jb    short chs_write_sectors
3220                                <1>       ;je    short chs_read_sectors
3221                                <1>       ;; (# of sectors to read is more than remaining sectors on the track)
3222                                <1>       ;mov   al, [sector_count]
3223                                <1> ;chs_read_sectors: ; read or write !
3224 000125D7 B001                  <1>       mov    al, 1 ; 25/02/2016
3225 000125D9 8A25[5D890100]        <1>       mov    ah, [disk_rw_op]  ; 02h = chs read, 03h = chs write
3226                                <1>       ;
3227 000125DF E87F2CFFFF            <1>       call   int13h             ; BIOS Service func ( ah ) = 2
3228                                <1>                                 ; Read disk sectors
3229                                <1>                                 ; AL-sec num CH-track CL-sec
3230                                <1>                                 ; DH-head DL-drive ES:BX-buffer
3231                                <1>                                 ; CF-flag AH-stat AL-sec read
3232                                <1>                                 ; If CF = 1 then (If AH > 0)
3233 000125E4 8825[5F890100]        <1>       mov    [disk_rw_err], ah
3234                                <1>
3235 000125EA 59                    <1>       pop    ecx
3236 000125EB 58                    <1>       pop    eax
3237 000125EC 7314                  <1>       jnc    short chs_read_ok
3238                                <1>
3239 000125EE 803D[5F890100]09      <1>       cmp    byte [disk_rw_err], 09h ; DMA crossed 64K segment boundary
3240 000125F5 7408                  <1>       je     short chs_read_error_retn
3241                                <1>
3242 000125F7 FE0D[5E890100]        <1>       dec    byte [retry_count]
3243 000125FD 75B0                  <1>       jnz    short chs_read_retry
3244                                <1>
3245                                <1> chs_read_error_retn:
3246 000125FF F9                    <1>       stc
3247                                <1>       ;retn
3248 00012600 EB69                  <1>       jmp    short update_drv_error_byte
3249                                <1>
3250                                <1> ;chs_write_sectors: ; read or write
3251                                <1>       ;; (# of sectors to read is less than remaining sectors on the track)
3252                                <1>       ;mov   [sector_count], al
3253                                <1>       ;jmp   short chs_read_sectors
3254                                <1>
3255                                <1> chs_read_ok:
3256                                <1>       ;; 23/02/2016
3257                                <1>       ;movzx edx, byte [sector_count] ; sector count (<= spt)
3258                                <1>       ;sub   ecx, edx  ; remaining sector count
3259                                <1>       ;jna   short update_drv_error_byte
3260                                <1>       ;add   eax, edx ; next disk sector
3261                                <1>       ;shl   edx, 9 ; 512 * sector count
3262                                <1>       ;add   ebx, edx ; next buffer byte address
3263                                <1>       ;jmp  chs_read_next_sector
3264                                <1>       ; 25/02/2016
3265 00012602 40                    <1>       inc    eax ; next sector
3266 00012603 81C300020000          <1>       add    ebx, 512
3267 00012609 E29D                  <1>       loop   chs_read_next_sector
3268 0001260B EB5E                  <1>       jmp    short update_drv_error_byte
3269                                <1>
3270                                <1> lba_write:
3271                                <1>       ; 23/02/2016
3272 0001260D C605[5D890100]1C      <1>       mov    byte [disk_rw_op], 1Ch ; LBA write
3273 00012614 EB07                  <1>       jmp    short lba_rw
3274                                <1>
3275                                <1> lba_read:
3276                                <1>       ; 23/02/2016
3277                                <1>       ; 17/02/2016
3278                                <1>       ; 14/02/2016
3279                                <1>       ; 13/02/2016
3280                                <1>       ; 31/01/2016 (TRDOS 386 =  TRDOS v2.0)
3281                                <1>       ; 10/07/2015 (Retro UNIX 386 v1)
3282                                <1>       ;
3283                                <1>       ; INPUT -> EAX = Logical Block Address
3284                                <1>       ;         ESI = Logical Dos Disk Table Offset (DRV)
3285                                <1>       ;         ECX = Sector Count
3286                                <1>       ;         EBX = Destination Buffer
3287                                <1>       ; OUTPUT ->
3288                                <1>       ;         cf = 0 or cf = 1
3289                                <1>       ; (Modified registers: EAX, EBX, ECX, EDX)
3290                                <1>
3291                                <1>       ; LBA read/write (with private LBA function)
3292                                <1>       ;((Retro UNIX 386 v1 - DISK I/O code by Erdogan Tan))
3293                                <1>
3294                                <1>
3295                                <1>       ; 23/02/2016
3296 00012616 C605[5D890100]1B      <1>       mov    byte [disk_rw_op], 1Bh ; LBA read
3297                                <1>
3298                                <1> lba_rw:
3299                                <1>       ; 17/02/2016
3300 0001261D 57                    <1>       push   edi
3301                                <1>
3302 0001261E 890D[60890100]        <1>       mov    [sector_count], ecx ; total sector (read) count
3303                                <1>
3304 00012624 8A5602                <1>       mov    dl, [esi+LD_PhyDrvNo]
3305                                <1>       ; dl = physical drive number (0,1, 80h, 81h, 82h, 83h)
3306                                <1>
3307                                <1> lba_read_next:
3308 00012627 81F900010000          <1>       cmp    ecx, 256
3309 0001262D 7605                  <1>       jna    short lba_read_rsc
3310 0001262F B900010000            <1>       mov    ecx, 256 ; 17/02/2016
3311                                <1> lba_read_rsc:
3312 00012634 290D[60890100]        <1>       sub    [sector_count], ecx ; remain sectors
3313                                <1>
3314 0001263A 89CF                  <1>       mov    edi, ecx
3315 0001263C 89C1                  <1>       mov    ecx, eax ; sector number/address
3316                                <1>
3317 0001263E C605[5E890100]04      <1>       mov    byte [retry_count], 4
3318                                <1> lba_read_retry:
3319 00012645 89F8                  <1>       mov    eax, edi
3320                                <1>       ;
3321                                <1>       ; ecx = sector number
3322                                <1>       ; al = sector count (0 - 255) /// (0 = 256)
3323                                <1>       ; dl = drive number
```

```
                                <1>       ; ebx = buffer offset
                                <1>       ;
                                <1>       ; Function 1Bh = LBA read, 1Ch = LBA write
                                <1>       ; 23/02/2016
00012647 8A25[5D890100]         <1>       mov    ah, [disk_rw_op] ; 1Bh = LBA read, 1Ch = LBA write
0001264D E8112CFFFF             <1>       call   int13h
                                <1>       ; al = ? (changed)
                                <1>       ; ah = error code
00012652 8825[5F890100]         <1>       mov    [disk_rw_err], ah
00012658 7334                   <1>       jnc    short lba_read_ok
0001265A 80FC80                 <1>       cmp    ah, 80h ; time out?
0001265D 740A                   <1>       je    short lba_read_stc_retn
0001265F FE0D[5E890100]         <1>       dec    byte [retry_count]
00012665 7FDE                   <1>       jg     short lba_read_retry
00012667 743A                   <1>       jz     short lba_read_reset
                                <1>       ; sf =  1
                                <1>
                                <1> lba_read_stc_retn:
00012669 F9                     <1>       stc
                                <1> lba_read_retn:
0001266A 5F                     <1>       pop    edi
                                <1>
                                <1> update_drv_error_byte:
0001266B 9C                     <1>       pushf
0001266C 53                     <1>       push   ebx
0001266D 6651                   <1>       push   cx
                                <1>       ;or    ecx, ecx
                                <1>       ;jz    short udrv_errb0
0001266F 8A0D[5F890100]         <1>       mov    cl, [disk_rw_err]
                                <1> udrv_errb0:
00012675 0FB65E02               <1>       movzx  ebx, byte [esi+LD_PhyDrvNo]
00012679 80FB02                 <1>       cmp    bl, 2
0001267C 7203                   <1>       jb    short udrv_errb1
0001267E 80EB7E                 <1>       sub    bl, 7Eh
                                <1>       ;cmp   bl, 5
                                <1>       ;ja    short udrv_errb2
                                <1> udrv_errb1:
00012681 81C3[41690000]         <1>       add    ebx, drv.error ; 13/02/2016
00012687 880B                   <1>       mov    [ebx], cl ; error code
                                <1> udrv_errb2:
00012689 6659                   <1>       pop    cx
0001268B 5B                     <1>       pop    ebx
0001268C 9D                     <1>       popf
0001268D C3                     <1>       retn
                                <1>
                                <1> lba_read_ok:
0001268E 89C8                   <1>       mov    eax, ecx ; sector number
00012690 01F8                   <1>       add    eax, edi ; sector number (next)
00012692 C1E709                 <1>       shl    edi, 9 ; sector count * 512
00012695 01FB                   <1>       add    ebx, edi ; next buffer offset
                                <1>
00012697 8B0D[60890100]         <1>       mov    ecx, [sector_count] ; remaining sectors
0001269D 09C9                   <1>       or     ecx, ecx
0001269F 7586                   <1>       jnz    short lba_read_next
000126A1 EBC7                   <1>       jmp    short lba_read_retn
                                <1>
                                <1> lba_read_reset:
000126A3 B40D                   <1>       mov    ah, 0Dh ; Alternate reset
000126A5 E8B92BFFFF             <1>       call   int13h
                                <1>       ; al = ? (changed)
                                <1>       ; ah = error code
000126AA 7399                   <1>       jnc    short lba_read_retry
000126AC EBBC                   <1>       jmp    short lba_read_retn
                                    %include 'trdosk8.s' ; 24/01/2016
                                <1> ; ****************************************************************************
                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.2) - MAIN PROGRAM : trdosk8.s
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Last Update: 17/04/2021
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Beginning: 24/01/2016
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Assembler: NASM version 2.11 (trdos386.s)
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
                                <1> ; u0.s (20/11/2015), u4.s (14/10/2015)
                                <1> ; ****************************************************************************
                                <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
                                <1> ; TRDOS2.ASM (09/11/2011)
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; DIR.ASM (c) 2004-2011 Erdogan TAN  [07/01/2004] Last Update: 09/10/2011
                                <1>
                                <1> set_run_sequence:
                                <1>       ; 23/12/2016
                                <1>       ; 10/06/2016
                                <1>       ; 22/05/2016
                                <1>       ; 20/05/2016
                                <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; TRDOS 386 feature only !
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     AL = process number (next process)
                                <1>       ;
                                <1>       ;     this process must be added to run sequence
                                <1>       ;
                                <1>       ;     [u.pri] = priority of present process
                                <1>       ;
                                <1>       ;     DL = priority (queue)
                                <1>       ;          0 = background (low) ; run on background
                                <1>       ;          1 = regular (normal) ; run as regular
                                <1>       ;          2 = event (high)     ; run for event
                                <1>       ;
                                <1>       ;     1) If the requested process is already running:
                                <1>       ;        a) If present priority is high ([u.pri]=2)
                                <1>       ;           and requested priority is also high,
                                <1>       ;           there is nothing to do! Because it has been
```

```
                        <1>       ;           done already (before this attempt).
                        <1>       ;        b) If present priority is high ([u.pri]=2)
                        <1>       ;           and requested priority is not high, there is
                        <1>       ;           nothing to do! Because, it's current
                        <1>       ;           run queue is unspecified, here. (It may be in
                        <1>       ;           a waiting list or in a run queue; if the new
                        <1>       ;           priority would be used to add it to relavant
                        <1>       ;             run queue, this would be wrong, unnecessary
                        <1>       ;           and destabilizing duplication!)
                        <1>       ;        c) If present priority is not high ([u.pri]<2)
                        <1>       ;             and requested priority is high (event),
                        <1>       ;           process will be added to present priority's
                        <1>       ;           run queue and then, priority will be changed
                        <1>       ;           to high ([u.pri]=2).
                        <1>       ;        d) If present priority is not high ([u.pri]<2)
                        <1>       ;           and requested priority is not high, [u.pri]
                        <1>       ;           value will be changed. There is nothing to do
                        <1>       ;           in addition. (The new priority value will be
                        <1>       ;           used by 'tswap/tswitch' procedure at 'sysret'
                        <1>       ;           or 'sysrele' stage.)
                        <1>       ;
                        <1>       ;     2) If the requested process is not running:
                        <1>       ;        a) If requested priority of the requested
                        <1>       ;           (next) process is high (event) and priority
                        <1>       ;           of present process is not high, the requested
                        <1>       ;           process will be added to ('runq_event') high
                        <1>       ;           priority run queue and then present (running)
                        <1>       ;           process will be stopped (swapped/switched out)
                        <1>       ;           immediately if it is in user mode, or it's
                        <1>       ;           [u.quant] value will be reset to 0 and (then)
                        <1>       ;           it will be stopped at 'sysret' stage.
                        <1>       ;        b) If requested priority of the requested
                        <1>       ;           (next) process is high (event) and priority
                        <1>       ;           of present process is also high, the requested
                        <1>       ;           process will be added to ('runq_event') high
                        <1>       ;           priority run queue and present (running)
                        <1>       ;           process will be allowed to run until it's
                        <1>       ;           time quantum will be elapsed ([u.quant]=0).
                        <1>       ;        c) If requested priority of the requested
                        <1>       ;           (next) process is not high ('run for event'),
                        <1>       ;           there is nothing to do. Because, it's current
                        <1>       ;           run queue is unspecified, here. (It may be in
                        <1>       ;           a waiting list or in a run queue; if the new
                        <1>       ;           priority would be used to add it to relavant
                        <1>       ;             run queue, this would be wrong, unnecessary
                        <1>       ;           and destabilizing duplication!)
                        <1>       ;
                        <1>       ; OUTPUT ->
                        <1>       ;     none
                        <1>       ;
                        <1>       ;     [u.pri] = priority of present process
                        <1>       ;
                        <1>       ;     cf = 1, if the request could not be fulfilled.
                        <1>       ;
                        <1>       ;     NOTE:
                        <1>       ;          * Processes in 'run as regular' queue can run
                        <1>       ;            if there is no process in 'run for event' queue
                        <1>       ;            ('run for event' processes have higher priority)
                        <1>       ;          * When [u.quant] time quantum of a process is
                        <1>       ;            elapsed, it's high priority ('run for event')
                        <1>       ;            status will be disabled, it can be run in sequence
                        <1>       ;            of it's actual run queue.
                        <1>       ;          * A 'run on background' process will always be
                        <1>       ;            sequenced in 'run on background' (low priority)
                        <1>       ;            queue, it can run only when other priority queues
                        <1>       ;            are empty. (idle time processes, e.g. printing)
                        <1>       ;
                        <1>       ; Modified registers: eax, ebx, edx
                        <1>       ;
                        <1>
                        <1> srunseq_0:
000126AE 3A05[B3030300] <1>       cmp   al, [u.uno]     ; same process ?
000126B4 750C           <1>       jne   short srunseq_2 ; no
                        <1>
000126B6 8A25[A9030300] <1>       mov   ah, [u.pri]  ; present/current priority
000126BC 80FC02         <1>       cmp   ah, 2        ; 'run for event' priority level
000126BF 7221           <1>       jb    short srunseq_6 ; no
                        <1>
                        <1> srunseq_1:
                        <1>       ; there is nothing to do!
000126C1 C3             <1>       retn
                        <1>
                        <1> srunseq_2:
                        <1>       ;;this not necessary ! 23/12/2016
                        <1>       ;;cmp  al, nproc    ; number of processes = 16
                        <1>       ;;jnb  short srunseq_5     ; error ! invalid process number
                        <1>
                        <1>       ; dl = priority
000126C2 80FA02         <1>       cmp   dl, 2        ; event queue
000126C5 72FA           <1>       jb    short srunseq_1 ; requested process is not present
                        <1>                             ; process and priority of requested
                        <1>                             ; process is not high (event),
                        <1>                             ; there is nothing to do!
                        <1>
                        <1>       ; requested process is not present process
                        <1>       ; & priority of requested process is high
000126C7 3A15[A9030300] <1>       cmp   dl, [u.pri]  ; priority of present process
000126CD 7606           <1>       jna   short srunseq_3 ; is high, also
                        <1>       ;
                        <1>       ; present process will be swapped/switched out
000126CF FE05[398D0100] <1>       inc   byte [p_change] ; 1
                        <1>
                        <1> srunseq_3:
                        <1>       ; add process to 'runq_event' queue for new event
000126D5 BB[52030300]   <1>       mov   ebx, runq_event ; high priority run queue
```

```
                                <1>
                                <1> srunseq_4:
                                <1>       ; al = process number
                                <1>       ; ebx = run queue
000126DA E887EDFFFF             <1>       call   putlu
000126DF C3                     <1>       retn
                                <1>
                                <1> srunseq_5:
000126E0 F5                     <1>       cmc
000126E1 C3                     <1>       retn
                                <1>
                                <1> srunseq_6:
                                <1>       ; present priority of the process is not high
                                <1>
000126E2 8815[A9030300]         <1>       mov    [u.pri], dl ; new priority
                                <1>                          ; (will be used by 'tswap')
                                <1>
000126E8 80FA02                 <1>       cmp    dl, 2        ; high priority ?
000126EB 72F3                   <1>       jb     short srunseq_5 ; no, there is nothing to do
                                <1>                            ; in addition
                                <1>
                                <1>       ; process must be added to relevant run queue, here!
                                <1>       ; (new priority is high/event priority and process
                                <1>       ; will not be added to a run queue by 'tswap')
                                <1>
000126ED BB[54030300]           <1>       mov    ebx, runq_normal ; 'run as regular' queue
                                <1>
000126F2 20E4                   <1>       and    ah, ah  ;  previous value of [u.pri]
000126F4 75E4                   <1>       jnz    short srunseq_4
                                <1>
000126F6 43                     <1>       inc    ebx
000126F7 43                     <1>       inc    ebx
                                <1>       ; ebx = runq_background ; 'run on backgroud' queue
                                <1>
000126F8 EBE0                   <1>       jmp    short srunseq_4
                                <1> clock:
                                <1>       ; 23/05/2016
                                <1>       ; 22/05/2016
                                <1>       ; 20/05/2016
                                <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; 14/05/2015 - 14/10/2015 (Retro UNIX 386 v1)
                                <1>       ; 07/12/2013 - 10/04/2014 (Retro UNIX 8086 v1)
                                <1>
000126FA 803D[A8030300]00       <1>       cmp    byte [u.quant], 0
00012701 772C                   <1>       ja     short clk_1
                                <1>       ;
00012703 803D[B3030300]01       <1>       cmp    byte [u.uno], 1 ; /etc/init ? (for Retro UNIX 8086 & 386 v1)
                                <1>                            ; MainProg (Kernel's Command Interpreter)
                                <1>                            ; for TRDOS 386.
0001270A 7623                   <1>       jna    short clk_1 ; yes, do not swap out
                                <1>       ;
0001270C 803D[5B030300]FF       <1>       cmp     byte [sysflg], 0FFh ; user or system space ?
00012713 7520                   <1>       jne    short clk_2      ; system space (sysflg <> 0FFh)
                                <1>       ;
00012715 66833D[AA030300]00     <1>       cmp    word [u.intr], 0
0001271D 7616                   <1>       jna    short clk_2
                                <1>       ;
                                <1>       ; 23/05/2016
0001271F 803D[3A8D0100]00       <1>       cmp    byte [multi_tasking], 0
00012726 760D                   <1>       jna    short clk_2
                                <1>       ;
00012728 FE05[398D0100]         <1>       inc    byte [p_change] ; it is time to change running process
0001272E C3                     <1>       retn
                                <1> clk_1:
0001272F FE0D[A8030300]         <1>       dec    byte [u.quant]
                                <1> clk_2:
00012735 C3                     <1>       retn   ; return to (hardware) timer interrupt routine
                                <1>
                                <1> ; 12/10/2017
                                <1> ; 15/01/2017
                                <1> ; 14/01/2017
                                <1> ; 07/01/2017
                                <1> ; 02/01/2017
                                <1> ; 17/08/2016
                                <1> ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                                <1> int34h:  ; #IOCTL# (I/O port access support for ring 3)
                                <1> ; 23/05/2016
                                <1>       ; 20/06/2016
                                <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     AH = 0 -> read port  (physical IO port) -byte-
                                <1>       ;     AH = 1 -> write port (physical IO port) -byte-
                                <1>       ;           AL = data byte
                                <1>       ;     AH = 2 -> read port  (physical IO port) -word-
                                <1>       ;     AH = 3 -> write port (physical IO port) -word-
                                <1>       ;           BX = data word
                                <1>       ;     AH = 4 -> read port  (physical IO port) -dword-
                                <1>       ;     AH = 5 -> write port (physical IO port) -dword-
                                <1>       ;           EBX = data dword
                                <1>       ;     ; 12/10/2017
                                <1>       ;     AH = 6 -> read port (physical IO port) twice -byte-
                                <1>       ;     AH = 7 -> write port (physical IO port) twice -byte-
                                <1>       ;           BX = data word
                                <1>       ;
                                <1>       ;     DX = Port number (<= 0FFFFh)
                                <1>       ;
                                <1>       ; OUTPUT ->
                                <1>       ;     AL = data byte (in al, dx)
                                <1>       ;     AX = data word (in ax, dx)
                                <1>       ;     EAX = data dword (in eax, dx)
                                <1>       ;
                                <1>       ;     (ECX = actual TRANSFER COUNT for string functions)
                                <1>       ;
                                <1>       ;
```

```
3365                            <1>       ; Modified registers: EAX
3366                            <1>       ;
3367                            <1>
3368                            <1>       ;cmp   ah, 5
3369                            <1>       ;ja    short int34h_5 ; invalid function !
3370                            <1>
3371                            <1>       ; 12/10/2017
3372 00012736 80FC07           <1>       cmp    ah, 7
3373 00012739 7743             <1>       ja     short int34h_5 ; invalid function !
3374                            <1>
3375                            <1>       ;; 15/01/2017
3376                            <1>       ; 14/01/2017
3377                            <1>       ; 02/01/2017
3378                            <1>       ;;mov  byte [ss:intflg], 34h     ; IOCTL interrupt
3379 0001273B FB               <1>       sti
3380                            <1>
3381                            <1>       ;sti   ; enable interrupts
3382 0001273C 80642408FE       <1>       and    byte [esp+8], 11111110b   ; clear carry bit of eflags register
3383                            <1>
3384 00012741 80FC01           <1>       cmp    ah, 1
3385 00012744 7205             <1>       jb     short int34h_0
3386 00012746 7705             <1>       ja     short int34h_1
3387                            <1>
3388 00012748 EE               <1>       out    dx, al
3389                            <1>       ;iretd
3390 00012749 EB01             <1>       jmp short int34h_iret
3391                            <1>
3392                            <1> int34h_0:
3393 0001274B EC               <1>       in     al, dx
3394                            <1>       ;iretd
3395                            <1> int34h_iret:
3396                            <1>       ;cli   ; 07/01/2017
3397                            <1>       ;; 15/01/2017
3398                            <1>       ;;mov  byte [ss:intflg], 0 ; reset
3399 0001274C CF               <1>       iretd
3400                            <1>
3401                            <1> int34h_1:
3402 0001274D F6C401           <1>       test   ah, 1
3403 00012750 7516             <1>       jnz    short int34h_3 ; out
3404                            <1>
3405                            <1>       ; in
3406 00012752 80FC02           <1>       cmp    ah, 2
3407 00012755 7707             <1>       ja     short int34h_2
3408                            <1>
3409 00012757 6689D8           <1>       mov    ax, bx
3410 0001275A 66ED             <1>       in     ax, dx
3411                            <1>       ;iretd
3412 0001275C EBEE             <1>       jmp    short int34h_iret
3413                            <1>
3414                            <1> int34h_2:
3415 0001275E 80FC04           <1>       cmp    ah, 4
3416 00012761 772C             <1>       ja     short int34h_7      ; 12/10/2017
3417                            <1>       ; ah = 4
3418 00012763 89D8             <1>       mov    eax, ebx
3419 00012765 ED               <1>       in     eax, dx
3420                            <1>       ;iretd
3421 00012766 EBE4             <1>       jmp    short int34h_iret
3422                            <1>
3423                            <1> int34h_3:
3424 00012768 80FC03           <1>       cmp    ah, 3
3425 0001276B 7707             <1>       ja     short int34h_4
3426                            <1>
3427 0001276D 6689D8           <1>       mov    ax, bx
3428 00012770 66EF             <1>       out    dx, ax
3429                            <1>       ;iretd
3430 00012772 EBD8             <1>       jmp    short int34h_iret
3431                            <1>
3432                            <1> int34h_4:
3433 00012774 80FC05           <1>       cmp    ah, 5
3434 00012777 770B             <1>       ja     short int34h_6      ; 12/10/2017
3435                            <1>       ; ah = 5
3436 00012779 89D8             <1>       mov    eax, ebx
3437 0001277B EF               <1>       out    dx, eax
3438                            <1>       ;iretd
3439 0001277C EBCE             <1>       jmp    short int34h_iret
3440                            <1>
3441                            <1> int34h_5:
3442 0001277E 804C240801       <1>       or     byte [esp+8], 1     ; set carry bit of eflags register
3443 00012783 CF               <1>       iretd
3444                            <1>
3445                            <1>       ; 12/10/2017
3446                            <1> int34h_6:
3447 00012784 6689D8           <1>       mov    ax, bx
3448 00012787 EE               <1>       out    dx, al
3449 00012788 EB00             <1>       jmp    short $+2
3450 0001278A 86E0             <1>       xchg   ah, al
3451 0001278C EE               <1>       out    dx, al
3452                            <1>       ;xchg  al, ah
3453                            <1>       ;iretd
3454 0001278D EB06             <1>       jmp    short int34h_8
3455                            <1> int34h_7:
3456 0001278F EC               <1>       in     al, dx
3457 00012790 EB00             <1>       jmp    short $+2
3458 00012792 88C4             <1>       mov    ah, al
3459 00012794 EC               <1>       in     al, dx
3460                            <1> int34h_8:
3461 00012795 86C4             <1>       xchg   al, ah
3462 00012797 CF               <1>       iretd
3463                            <1>
3464                            <1>
3465                            <1> INT4Ah:
3466                            <1>       ; 24/01/2016
3467                            <1>       ; this procedure will be called by 'RTC_INT' (in 'timer.s')
3468 00012798 C3               <1>       retn
3469                            <1>
```

```
                                <1> ; u0.s
                                <1> ; Retro UNIX 386 v1 Kernel (v0.2) - SYS0.INC
                                <1> ; Last Modification: 20/11/2015
                                <1>
                                <1> com2_int:
                                <1>       ; 07/11/2015
                                <1>       ; 24/10/2015
                                <1>       ; 23/10/2015
                                <1>       ; 14/03/2015 (Retro UNIX 386 v1 - Beginning)
                                <1>       ; 28/07/2014 (Retro UNIX 8086 v1)
                                <1>       ; < serial port 2 interrupt handler >
                                <1>       ;
00012799 890424                 <1>       mov    [esp], eax ; overwrite call return address
                                <1>       ;push  eax
0001279C 66B80900               <1>       mov    ax, 9
000127A0 EB07                   <1>       jmp    short comm_int
                                <1> com1_int:
                                <1>       ; 07/11/2015
                                <1>       ; 24/10/2015
000127A2 890424                 <1>       mov    [esp], eax ; overwrite call return address
                                <1>       ; 23/10/2015
                                <1>       ;push  eax
000127A5 66B80800               <1>       mov    ax, 8
                                <1> comm_int:
                                <1>       ; 20/11/2015
                                <1>       ; 18/11/2015
                                <1>       ; 17/11/2015
                                <1>       ; 16/11/2015
                                <1>       ; 09/11/2015
                                <1>       ; 08/11/2015
                                <1>       ; 07/11/2015
                                <1>       ; 06/11/2015 (serial4.asm, 'serial')
                                <1>       ; 01/11/2015
                                <1>       ; 26/10/2015
                                <1>       ; 23/10/2015
000127A9 53                     <1>       push   ebx
000127AA 56                     <1>       push   esi
000127AB 57                     <1>       push   edi
000127AC 1E                     <1>       push   ds
000127AD 06                     <1>       push   es
                                <1>       ; 18/11/2015
000127AE 0F20DB                 <1>       mov    ebx, cr3
000127B1 53                     <1>       push   ebx ; ****
                                <1>       ;
000127B2 51                     <1>       push   ecx ; ***
000127B3 52                     <1>       push   edx ; **
                                <1>       ;
000127B4 BB10000000             <1>       mov    ebx, KDATA
000127B9 8EDB                   <1>       mov    ds, bx
000127BB 8EC3                   <1>       mov    es, bx
                                <1>       ;
000127BD 8B0D[A8800100]         <1>       mov    ecx, [k_page_dir]
000127C3 0F22D9                 <1>       mov    cr3, ecx
                                <1>       ; 20/11/2015
                                <1>       ; Interrupt identification register
000127C6 66BAFA02               <1>       mov    dx, 2FAh ; COM2
                                <1>       ;
000127CA 3C08                   <1>       cmp    al, 8
000127CC 7702                   <1>       ja     short com_i0
                                <1>       ;
                                <1>       ; 20/11/2015
                                <1>       ; 17/11/2015
                                <1>       ; 16/11/2015
                                <1>       ; 15/11/2015
                                <1>       ; 24/10/2015
                                <1>       ; 14/03/2015 (Retro UNIX 386 v1 - Beginning)
                                <1>       ; 28/07/2014 (Retro UNIX 8086 v1)
                                <1>       ; < serial port 1 interrupt handler >
                                <1>       ;
000127CE FEC6                   <1>       inc    dh ; 3FAh ; COM1 Interrupt id. register
                                <1> com_i0:
                                <1>       ;push  eax ; *
                                <1>       ; 07/11/2015
000127D0 A2[12810100]           <1>       mov    byte [ccomport], al
                                <1>       ; 09/11/2015
000127D5 0FB7D8                 <1>       movzx  ebx, ax ; 8 or 9
                                <1>       ; 17/11/2015
                                <1>       ; reset request for response status
000127D8 88A3[08810100]         <1>       mov    [ebx+req_resp-8], ah ; 0
                                <1>       ;
                                <1>       ; 20/11/2015
000127DE EC                     <1>       in     al, dx       ; read interrupt id. register
000127DF EB00                   <1>       JMP    $+2          ; I/O DELAY
000127E1 2404                   <1>       and    al, 4        ; received data available?
000127E3 7470                   <1>       jz     short com_eoi; (transmit. holding reg. empty)
                                <1>       ;
                                <1>       ; 20/11/2015
000127E5 80EA02                 <1>       sub    dl, 3FAh-3F8h; data register (3F8h, 2F8h)
000127E8 EC                     <1>       in     al, dx       ; read character
                                <1>       ;JMP   $+2          ; I/O DELAY
                                <1>       ; 08/11/2015
                                <1>       ; 07/11/2015
000127E9 89DE                   <1>       mov    esi, ebx
000127EB 89DF                   <1>       mov    edi, ebx
000127ED 81C6[0C810100]         <1>       add    esi, rchar - 8 ; points to last received char
000127F3 81C7[0E810100]         <1>       add    edi, schar - 8 ; points to last sent char
000127F9 8806                   <1>       mov    [esi], al ; received char (current char)
                                <1>       ; query
000127FB 20C0                   <1>       and    al, al
000127FD 7527                   <1>       jnz    short com_i2
                                <1>       ; response
                                <1>       ; 17/11/2015
                                <1>       ; set request for response status
000127FF FE83[08810100]         <1>       inc    byte [ebx+req_resp-8] ; 1
                                <1>       ;
```

```
3575 00012805 6683C205          <1>       add    dx, 3FDh-3F8h; (3FDh, 2FDh)
3576 00012809 EC                <1>       in     al, dx       ; read line status register
3577 0001280A EB00              <1>       JMP    $+2          ; I/O DELAY
3578 0001280C 2420              <1>       and    al, 20h            ; transmitter holding reg. empty?
3579 0001280E 7445              <1>       jz     short com_eoi      ; no
3580 00012810 B0FF              <1>       mov    al, 0FFh     ; response
3581 00012812 6683EA05          <1>       sub    dx, 3FDh-3F8h      ; data port (3F8h, 2F8h)
3582 00012816 EE                <1>       out    dx, al       ; send on serial port
3583                            <1>       ; 17/11/2015
3584 00012817 803F00            <1>       cmp    byte [edi], 0   ; query ? (schar)
3585 0001281A 7502              <1>       jne    short com_i1    ; no
3586 0001281C 8807              <1>       mov    [edi], al    ; 0FFh (responded)
3587                            <1> com_i1:
3588                            <1>       ; 17/11/2015
3589                            <1>       ; reset request for response status (again)
3590 0001281E FE8B[08810100]    <1>         dec     byte [ebx+req_resp-8] ; 0
3591 00012824 EB2F              <1>       jmp    short com_eoi
3592                            <1> com_i2:
3593                            <1>       ; 08/11/2015
3594 00012826 3CFF              <1>       cmp    al, 0FFh     ; (response ?)
3595 00012828 7417              <1>       je     short com_i3 ; (check for response signal)
3596                            <1>       ; 07/11/2015
3597 0001282A 3C04              <1>       cmp    al, 04h      ; EOT
3598 0001282C 751C              <1>       jne    short com_i4
3599                            <1>       ; EOT = 04h (End of Transmit) - 'CTRL + D'
3600                            <1>       ;(an EOT char is supposed as a ctrl+brk from the terminal)
3601                            <1>       ; 08/11/2015
3602                            <1>             ; ptty -> tty 0 to 7 (pseudo screens)
3603 0001282E 861D[D6800100]    <1>       xchg   bl, [ptty]  ; tty number (8 or 9)
3604 00012834 E89748FFFF        <1>       call   ctrlbrk
3605 00012839 861D[D6800100]    <1>       xchg   [ptty], bl ; (restore ptty value and BL value)
3606                            <1>       ;mov   al, 04h ; EOT
3607                            <1>       ; 08/11/2015
3608 0001283F EB09              <1>       jmp    short com_i4
3609                            <1> com_i3:
3610                            <1>       ; 08/11/2015
3611                            <1>       ; If 0FFh has been received just after a query
3612                            <1>       ; (schar, ZERO), it is a response signal.
3613                            <1>       ; 17/11/2015
3614 00012841 803F00            <1>       cmp    byte [edi], 0 ; query ? (schar)
3615 00012844 7704              <1>       ja     short com_i4 ; no
3616                            <1>       ; reset query status (schar)
3617 00012846 8807              <1>       mov    [edi], al ; 0FFh
3618 00012848 FEC0              <1>       inc    al ; 0
3619                            <1> com_i4:
3620                            <1>       ; 27/07/2014
3621                            <1>       ; 09/07/2014
3622 0001284A D0E3              <1>       shl    bl, 1
3623 0001284C 81C3[D8800100]    <1>       add    ebx, ttychr
3624                            <1>       ; 23/07/2014 (always overwrite)
3625                            <1>       ;;cmp  word [ebx], 0
3626                            <1>       ;;ja   short com_eoi
3627                            <1>       ;
3628 00012852 668903            <1>       mov    [ebx], ax   ; Save ascii code
3629                            <1>                          ; scan code = 0
3630                            <1> com_eoi:
3631                            <1>       ;mov   al, 20h
3632                            <1>       ;out   20h, al         ; end of interrupt
3633                            <1>       ;
3634                            <1>       ; 07/11/2015
3635                            <1>       ;pop   eax ; *
3636 00012855 A0[12810100]      <1>       mov    al, byte [ccomport] ; current COM port
3637                            <1        ; al = tty number (8 or 9)
3638 0001285A E85E010000        <1>       call   wakeup
3639                            <1> com_iret:
3640                            <1>       ; 23/10/2015
3641 0001285F 5A                <1>       pop    edx ; **
3642 00012860 59                <1>       pop    ecx ; ***
3643                            <1>       ; 18/11/2015
3644                            <1>       ;pop   eax ; ****
3645                            <1>       ;mov   cr3, eax
3646                            <1>       ;jmp   iiret
3647 00012861 E93DE5FEFF        <1>       jmp    iiretp
3648                            <1>
3649                            <1> ;iiretp: ; 01/09/2015
3650                            <1> ;     ; 28/08/2015
3651                            <1> ;     pop    eax ; (*) page directory
3652                            <1> ;     mov    cr3, eax
3653                            <1> ;iiret:
3654                            <1> ;     ; 22/08/2014
3655                            <1> ;     mov    al, 20h ; END OF INTERRUPT COMMAND TO 8259
3656                            <1> ;     out    20h, al      ; 8259 PORT
3657                            <1> ;     ;
3658                            <1> ;     pop    es
3659                            <1> ;     pop    ds
3660                            <1> ;     pop    edi
3661                            <1> ;     pop    esi
3662                            <1> ;     pop    ebx ; 29/08/2014
3663                            <1> ;     pop    eax
3664                            <1> ;     iretd
3665                            <1>
3666                            <1> sp_init:
3667                            <1>       ; 07/11/2015
3668                            <1>       ; 29/10/2015
3669                            <1>       ; 26/10/2015
3670                            <1>       ; 23/10/2015
3671                            <1>       ; 29/06/2015
3672                            <1>       ; 14/03/2015 (Retro UNIX 386 v1 - 115200 baud)
3673                            <1>       ; 28/07/2014 (Retro UNIX 8086 v1 - 9600 baud)
3674                            <1>       ; Initialization of Serial Port Communication Parameters
3675                            <1>       ; (COM1 base port address = 3F8h, COM1 Interrupt = IRQ 4)
3676                            <1>       ; (COM2 base port address = 2F8h, COM1 Interrupt = IRQ 3)
3677                            <1>       ;
3678                            <1>       ; ((Modified registers: EAX, ECX, EDX, EBX))
3679                            <1>       ;
```

```
                        <1>       ; INPUT:  (29/06/2015)
                        <1>       ;     AL = 0 for COM1
                        <1>       ;          1 for COM2
                        <1>       ;     AH = Communication parameters
                        <1>       ;
                        <1>       ;  (*) Communication parameters (except BAUD RATE):
                        <1>       ;     Bit    4      3      2      1      0
                        <1>       ;            -PARITY--   STOP BIT  -WORD LENGTH-
                        <1>       ;  this one -->     00 = none    0 = 1 bit  11 = 8 bits
                        <1>       ;            01 = odd     1 = 2 bits   10 = 7 bits
                        <1>       ;            11 = even
                        <1>       ;  Baud rate setting bits: (29/06/2015)
                        <1>       ;            Retro UNIX 386 v1 feature only !
                        <1>       ;     Bit    7    6    5  | Baud rate
                        <1>       ;            ------------------------
                        <1>       ;     value  0    0    0  | Default (Divisor = 1)
                        <1>       ;            0    0    1  | 9600 (12)
                        <1>       ;            0    1    0  | 19200 (6)
                        <1>       ;            0    1   1   | 38400 (3)
                        <1>       ;            1    0   0   | 14400 (8)
                        <1>       ;            1    0   1   | 28800 (4)
                        <1>       ;            1    1    0  | 57600 (2)
                        <1>       ;            1    1    1  | 115200 (1)
                        <1>
                        <1>       ; References:
                        <1>       ; (1) IBM PC-XT Model 286 BIOS Source Code
                        <1>       ;     RS232.ASM --- 10/06/1985 COMMUNICATIONS BIOS (RS232)
                        <1>       ; (2) Award BIOS 1999 - ATORGS.ASM
                        <1>       ; (3) http://wiki.osdev.org/Serial_Ports
                        <1>       ;
                        <1>       ; Set communication parameters for COM1 (= 03h)
                        <1>       ;
00012866 BB[0E810100]   <1>       mov    ebx, com1p          ; COM1 parameters
0001286B 66BAF803       <1>       mov    dx, 3F8h            ; COM1
                        <1>        ; 29/10/2015
0001286F 66B90103       <1>       mov    cx, 301h  ; divisor = 1 (115200 baud)
00012873 E86F000000     <1>       call   sp_i3  ; call A4
00012878 A880           <1>       test   al, 80h
0001287A 7410           <1>       jz     short sp_i0 ; OK..
                        <1>              ; Error !
                        <1>       ;mov   dx, 3F8h
0001287C 80EA05         <1>       sub    dl, 5 ; 3FDh -> 3F8h
0001287F 66B90E03       <1>       mov    cx, 30Eh  ; divisor = 12 (9600 baud)
00012883 E85F000000     <1>       call   sp_i3  ; call A4
00012888 A880           <1>       test   al, 80h
0001288A 7508           <1>       jnz    short sp_i1
                        <1> sp_i0:
                        <1>         ; (Note: Serial port interrupts will be disabled here...)
                        <1>         ; (INT 14h initialization code disables interrupts.)
                        <1>       ;
0001288C C603E3         <1>       mov    byte [ebx], 0E3h ; 11100011b
0001288F E8DC000000     <1>       call   sp_i5 ; 29/06/2015
                        <1> sp_i1:
00012894 43             <1>       inc    ebx
00012895 66BAF802       <1>       mov    dx, 2F8h            ; COM2
                        <1>        ; 29/10/2015
00012899 66B90103       <1>       mov    cx, 301h  ; divisor = 1 (115200 baud)
0001289D E845000000     <1>       call   sp_i3  ; call A4
000128A2 A880           <1>       test   al, 80h
000128A4 7410           <1>       jz     short sp_i2 ; OK..
                        <1>              ; Error !
                        <1>       ;mov   dx, 2F8h
000128A6 80EA05         <1>       sub    dl, 5 ; 2FDh -> 2F8h
000128A9 66B90E03       <1>       mov    cx, 30Eh  ; divisor = 12 (9600 baud)
000128AD E835000000     <1>       call   sp_i3  ; call A4
000128B2 A880           <1>       test   al, 80h
000128B4 7530           <1>       jnz    short sp_i7
                        <1> sp_i2:
000128B6 C603E3         <1>       mov    byte [ebx], 0E3h ; 11100011b
                        <1> sp_i6:
                        <1>       ;; COM2 - enabling IRQ 3
                        <1>       ; 07/11/2015
                        <1>       ; 26/10/2015
000128B9 9C             <1>       pushf
000128BA FA             <1>       cli
                        <1>       ;
000128BB 66BAFC02       <1>       mov    dx, 2FCh            ; modem control register
000128BF EC             <1>       in     al, dx                   ; read register
000128C0 EB00           <1>       JMP    $+2                 ; I/O DELAY
000128C2 0C08           <1>       or     al, 8               ; enable bit 3 (OUT2)
000128C4 EE             <1>       out    dx, al              ; write back to register
000128C5 EB00           <1>       JMP    $+2                 ; I/O DELAY
000128C7 66BAF902       <1>       mov    dx, 2F9h            ; interrupt enable register
000128CB EC             <1>       in     al, dx              ; read register
000128CC EB00           <1>       JMP    $+2                 ; I/O DELAY
                        <1>       ;or    al, 1               ; receiver data interrupt enable and
000128CE 0C03           <1>       or     al, 3               ; transmitter empty interrupt enable
000128D0 EE             <1>       out    dx, al                   ; write back to register
000128D1 EB00           <1>       JMP    $+2                 ; I/O DELAY
000128D3 E421           <1>       in     al, 21h             ; read interrupt mask register
000128D5 EB00           <1>       JMP    $+2                 ; I/O DELAY
000128D7 24F7           <1>       and    al, 0F7h            ; enable IRQ 3 (COM2)
000128D9 E621           <1>       out    21h, al             ; write back to register
                        <1>       ;
                        <1>       ; 23/10/2015
000128DB B8[99270100]   <1>       mov    eax, com2_int
000128E0 A3[B8290100]   <1>       mov    [com2_irq3], eax
                        <1>       ; 26/10/2015
000128E5 9D             <1>       popf
                        <1> sp_i7:
000128E6 C3             <1>       retn
                        <1>
                        <1> sp_i3:
                        <1> ;A4:        ;----- INITIALIZE THE COMMUNICATIONS PORT
                        <1>       ; 28/10/2015
```

```
3785 000128E7 FEC2              <1>       inc    dl     ; 3F9h (2F9h); 3F9h, COM1 Interrupt enable register
3786 000128E9 B000              <1>       mov    al, 0
3787 000128EB EE                <1>       out    dx, al              ; disable serial port interrupt
3788 000128EC EB00              <1>       JMP    $+2                 ; I/O DELAY
3789 000128EE 80C202            <1>       add    dl, 2  ; 3FBh (2FBh); COM1 Line control register (3FBh)
3790 000128F1 B080              <1>       mov    al, 80h
3791 000128F3 EE                <1>       out    dx, al              ; SET DLAB=1 ; divisor latch access bit
3792                            <1>       ;----- SET BAUD RATE DIVISOR
3793                            <1>       ; 26/10/2015
3794 000128F4 80EA03            <1>       sub    dl, 3   ; 3F8h (2F8h)     ; register for least significant byte
3795                            <1>                                        ; of the divisor value
3796 000128F7 88C8              <1>       mov    al, cl ; 1
3797 000128F9 EE                <1>       out    dx, al              ; 1 = 115200 baud (Retro UNIX 386 v1)
3798                            <1>                                  ; 2 = 57600 baud
3799                            <1>                                  ; 3 = 38400 baud
3800                            <1>                                  ; 6 = 19200 baud
3801                            <1>                                  ; 12 = 9600 baud (Retro UNIX 8086 v1)
3802 000128FA EB00              <1>       JMP    $+2                 ; I/O DELAY
3803 000128FC 28C0              <1>       sub    al, al
3804 000128FE FEC2              <1>       inc    dl      ; 3F9h (2F9h)     ; register for most significant byte
3805                            <1>                                        ; of the divisor value
3806 00012900 EE                <1>       out    dx, al ; 0
3807 00012901 EB00              <1>       JMP    $+2                 ; I/O DELAY
3808                            <1>       ;
3809 00012903 88E8              <1>       mov    al, ch ; 3          ; 8 data bits, 1 stop bit, no parity
3810                            <1>       ;and   al, 1Fh ; Bits 0,1,2,3,4
3811 00012905 80C202            <1>       add    dl, 2  ; 3FBh (2FBh); Line control register
3812 00012908 EE                <1>       out    dx, al
3813 00012909 EB00              <1>       JMP    $+2                 ; I/O DELAY
3814                            <1>       ; 29/10/2015
3815 0001290B FECA              <1>       dec    dl     ; 3FAh (2FAh); FIFO Control register (16550/16750)
3816 0001290D 30C0              <1>       xor    al, al              ; 0
3817 0001290F EE                <1>       out    dx, al              ; Disable FIFOs (reset to 8250 mode)
3818 00012910 EB00              <1>       JMP    $+2
3819                            <1> sp_i4:
3820                            <1> ;A18: ;----- COMM PORT STATUS ROUTINE
3821                            <1>       ; 29/06/2015 (line status after modem status)
3822 00012912 80C204            <1>       add    dl, 4  ; 3FEh (2FEh); Modem status register
3823                            <1> sp_i4s:
3824 00012915 EC                <1>       in     al, dx              ; GET MODEM CONTROL STATUS
3825 00012916 EB00              <1>       JMP    $+2                 ; I/O DELAY
3826 00012918 88C4              <1>       mov    ah, al              ; PUT IN (AH) FOR RETURN
3827 0001291A FECA              <1>       dec    dl     ; 3FDh (2FDh); POINT TO LINE STATUS REGISTER
3828                            <1>                                  ; dx = 3FDh for COM1, 2FDh for COM2
3829 0001291C EC                <1>       in     al, dx              ; GET LINE CONTROL STATUS
3830                            <1>       ; AL = Line status, AH = Modem status
3831 0001291D C3                <1>       retn
3832                            <1>
3833                            <1> sp_status:
3834                            <1>       ; 29/06/2015
3835                            <1>       ; 27/06/2015 (Retro UNIX 386 v1)
3836                            <1>       ; Get serial port status
3837 0001291E 66BAFE03          <1>       mov    dx, 3FEh            ; Modem status register (COM1)
3838 00012922 28C6              <1>       sub    dh, al              ; dh = 2 for COM2 (al = 1)
3839                            <1>                                  ; dx = 2FEh for COM2
3840 00012924 EBEF              <1>       jmp    short sp_i4s
3841                            <1>
3842                            <1> sp_setp: ; Set serial port communication parameters
3843                            <1>       ; 07/11/2015
3844                            <1>       ; 29/10/2015
3845                            <1>       ; 29/06/2015
3846                            <1>       ; Retro UNIX 386 v1 feature only !
3847                            <1>       ;
3848                            <1>       ; INPUT:
3849                            <1>       ;      AL = 0 for COM1
3850                            <1>       ;           1 for COM2
3851                            <1>       ;      AH = Communication parameters (*)
3852                            <1>       ; OUTPUT:
3853                            <1>       ;      CL = Line status
3854                            <1>       ;      CH = Modem status
3855                            <1>       ;   If cf = 1 -> Error code in [u.error]
3856                            <1>       ;              'invalid parameter !'
3857                            <1>       ;                   or
3858                            <1>       ;             'device not ready !' error
3859                            <1>       ;
3860                            <1>       ;  (*) Communication parameters (except BAUD RATE):
3861                            <1>       ;      Bit    4      3      2      1      0
3862                            <1>       ;             -PARITY--   STOP BIT  -WORD LENGTH-
3863                            <1>       ;  this one -->     00 = none    0 = 1 bit  11 = 8 bits
3864                            <1>       ;             01 = odd     1 = 2 bits   10 = 7 bits
3865                            <1>       ;             11 = even
3866                            <1>       ;  Baud rate setting bits: (29/06/2015)
3867                            <1>       ;             Retro UNIX 386 v1 feature only !
3868                            <1>       ;      Bit    7    6    5  | Baud rate
3869                            <1>       ;             ------------------------
3870                            <1>       ;      value  0    0    0  | Default (Divisor = 1)
3871                            <1>       ;             0    0    1  | 9600 (12)
3872                            <1>       ;             0    1    0  | 19200 (6)
3873                            <1>       ;             0    1   1   | 38400 (3)
3874                            <1>       ;             1    0   0   | 14400 (8)
3875                            <1>       ;             1    0   1   | 28800 (4)
3876                            <1>       ;             1    1    0  | 57600 (2)
3877                            <1>       ;             1    1    1  | 115200 (1)
3878                            <1>       ;
3879                            <1>       ; (COM1 base port address = 3F8h, COM1 Interrupt = IRQ 4)
3880                            <1>       ; (COM2 base port address = 2F8h, COM1 Interrupt = IRQ 3)
3881                            <1>       ;
3882                            <1>       ; ((Modified registers: EAX, ECX, EDX, EBX))
3883                            <1>       ;
3884 00012926 66BAF803          <1>       mov    dx, 3F8h
3885 0001292A BB[0E810100]      <1>       mov    ebx, com1p ; COM1 control byte offset
3886 0001292F 3C01              <1>       cmp    al, 1
3887 00012931 776B              <1>       ja     short sp_invp_err
3888 00012933 7203              <1>       jb     short sp_setp1 ;  COM1 (AL = 0)
3889 00012935 FECE              <1>       dec    dh ; 2F8h
```

```
3890 00012937 43                  <1>       inc    ebx ; COM2 control byte offset
3891                              <1> sp_setp1:
3892                              <1>       ; 29/10/2015
3893 00012938 8823                <1>       mov    [ebx], ah
3894 0001293A 0FB6CC              <1>       movzx  ecx, ah
3895 0001293D C0E905              <1>       shr    cl, 5 ; -> baud rate index
3896 00012940 80E41F              <1>       and    ah, 1Fh ; communication parameters except baud rate
3897 00012943 8A81[AD290100]      <1>       mov    al, [ecx+b_div_tbl]
3898 00012949 6689C1              <1>       mov    cx, ax
3899 0001294C E896FFFFFF          <1>       call   sp_i3
3900 00012951 6689C1              <1>       mov    cx, ax ; CL = Line status, CH = Modem status
3901 00012954 A880                <1>       test   al, 80h
3902 00012956 740F                <1>       jz     short sp_setp2
3903 00012958 C603E3              <1>       mov    byte [ebx], 0E3h ; Reset to initial value (11100011b)
3904                              <1> stp_dnr_err:
3905 0001295B C705[C8030300]0F00- <1>       mov    dword [u.error], ERR_DEV_NOT_RDY ; 'device not ready !'
3905 00012963 0000                <1>
3906                              <1>       ; CL = Line status, CH = Modem status
3907 00012965 F9                  <1>       stc
3908 00012966 C3                  <1>       retn
3909                              <1> sp_setp2:
3910 00012967 80FE02              <1>       cmp    dh, 2 ; COM2 (2F?h)
3911 0001296A 0F8649FFFFFF        <1>       jna    sp_i6
3912                              <1>                    ; COM1 (3F?h)
3913                              <1> sp_i5:
3914                              <1>       ; 07/11/2015
3915                              <1>       ; 26/10/2015
3916                              <1>       ; 29/06/2015
3917                              <1>       ;
3918                              <1>       ;; COM1 - enabling IRQ 4
3919 00012970 9C                  <1>       pushf
3920 00012971 FA                  <1>       cli
3921 00012972 66BAFC03            <1>       mov    dx, 3FCh            ; modem control register
3922 00012976 EC                  <1>       in     al, dx                    ; read register
3923 00012977 EB00                <1>       JMP    $+2                 ; I/O DELAY
3924 00012979 0C08                <1>       or     al, 8               ; enable bit 3 (OUT2)
3925 0001297B EE                  <1>       out    dx, al              ; write back to register
3926 0001297C EB00                <1>       JMP    $+2                 ; I/O DELAY
3927 0001297E 66BAF903            <1>       mov    dx, 3F9h            ; interrupt enable register
3928 00012982 EC                  <1>       in     al, dx              ; read register
3929 00012983 EB00                <1>       JMP    $+2                 ; I/O DELAY
3930                              <1>       ;or    al, 1               ; receiver data interrupt enable and
3931 00012985 0C03                <1>       or     al, 3               ; transmitter empty interrupt enable
3932 00012987 EE                  <1>       out    dx, al                    ; write back to register
3933 00012988 EB00                <1>       JMP    $+2                 ; I/O DELAY
3934 0001298A E421                <1>       in     al, 21h             ; read interrupt mask register
3935 0001298C EB00                <1>       JMP    $+2                 ; I/O DELAY
3936 0001298E 24EF                <1>       and    al, 0EFh            ; enable IRQ 4 (COM1)
3937 00012990 E621                <1>       out    21h, al             ; write back to register
3938                              <1>       ;
3939                              <1>       ; 23/10/2015
3940 00012992 B8[A2270100]        <1>       mov    eax, com1_int
3941 00012997 A3[B4290100]        <1>       mov    [com1_irq4], eax
3942                              <1>       ; 26/10/2015
3943 0001299C 9D                  <1>       popf
3944 0001299D C3                  <1>       retn
3945                              <1>
3946                              <1> sp_invp_err:
3947 0001299E C705[C8030300]1700- <1>       mov    dword [u.error], ERR_INV_PARAMETER ; 'invalid parameter !'
3947 000129A6 0000                <1>
3948 000129A8 31C9                <1>       xor    ecx, ecx
3949 000129AA 49                  <1>       dec    ecx ; 0FFFFh
3950 000129AB F9                  <1>       stc
3951 000129AC C3                  <1>       retn
3952                              <1>
3953                              <1> ; 29/10/2015
3954                              <1> b_div_tbl: ; Baud rate divisor table (115200/divisor)
3955 000129AD 010C0603080401      <1>       db 1, 12, 6, 3, 8, 4, 1
3956                              <1>
3957                              <1>
3958                              <1> ; 23/10/2015
3959                              <1> com1_irq4:
3960 000129B4 [BC290100]          <1>       dd dummy_retn
3961                              <1> com2_irq3:
3962 000129B8 [BC290100]          <1>       dd dummy_retn
3963                              <1>
3964                              <1> dummy_retn:
3965 000129BC C3                  <1>       retn
3966                              <1>
3967                              <1> wakeup:
3968                              <1>       ; 24/01/2016
3969 000129BD C3                  <1>       retn
3970                              <1>
3971                              <1> set_working_path_x:
3972                              <1>             ; 17/10/2016 (TRDOS 386 - FFF & FNF)
3973 000129BE 66B80100            <1>             mov    ax, 1
3974                              <1>                    ; File name is needed/forced (AL=1)
3975                              <1>                    ; Change directory as temporary (AH=0)
3976                              <1>
3977                              <1> set_working_path_xx: ; 30/12/2017 (syschdir)
3978                              <1>             ; This is needed for preventing wrong Find Next File
3979                              <1>             ; system call after sysopen, syscreate, sysmkdir etc.
3980                              <1>             ; Find Next File must immediate follow Find First File)
3981                              <1>
3982 000129C2 8825[5C8D0100]      <1>             mov    [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
3983                              <1>
3984                              <1> set_working_path:
3985                              <1>             ; 16/10/2016
3986                              <1>             ; 12/10/2016
3987                              <1>             ; 10/10/2016
3988                              <1>             ; 05/10/2016 - TRDOS 386 (TRDOS v2.0)
3989                              <1>             ;
3990                              <1>             ; TRDOS v1.0 (DIR.ASM, "proc_set_working_path")
3991                              <1>                ; 27/01/2011 - 08/02/2011
3992                              <1>             ; Set/Changes current drive, directory and file
```

```
                                <1>             ; depending on command tail
                                <1>             ; (procedure is derivated from CMD_INTR.ASM
                                <1>             ; file or dir locating code of internal commands)
                                <1>             ; (This procedure is prepared for INT 21H file/dir
                                <1>             ; functions and also to get compact code for
                                <1>             ; internal mainprog -command interpreter- commands)
                                <1>             ;
                                <1>             ; INPUT: DS:SI -> Command tail (ASCIIZ string)
                                <1>             ; AL = 0 -> any, AL > 0 -> file name is forced
                                <1>             ; AH = CD -> Change directory permanently
                                <1>             ; AH <> CD -> Change directory as temporary
                                <1>             ;
                                <1>             ; OUTPUT: ES=DS, FindFile structure has been set
                                <1>             ;       RUN_CDRV points previous current drive
                                <1>             ;       DS:SI = FindFile structure address
                                <1>             ;       (DS=CS)
                                <1>             ;       AX, BX, CX, DX, DI will be changed
                                <1>             ;   cf = 1 -> Error code in AX (AL)
                                <1>             ;       stc & AX = 0 -> Bad command or path name
                                <1>             ; ------------------------------------------------
                                <1>             ;
                                <1>             ; TRDOS 386 (05/10/2016)
                                <1>             ; INPUT:
                                <1>             ;     ESI = File/Directory Path (ASCIIZ string)
                                <1>             ;           address in user's memory space
                                <1>             ;      AL = 0 -> any
                                <1>             ;      AL > 0 -> file name is forced
                                <1>             ;      AH = CD -> change directory as permanent
                                <1>             ;      AH <> CD -> change directory as temporary
                                <1>             ;
                                <1>             ; OUTPUT:
                                <1>             ;     FindFile structure has been set
                                <1>             ;      RUN_CDRV points previous current drive
                                <1>             ;      ESI = FindFile_Name address ; 12/10/2016
                                <1>             ;
                                <1>             ;       cf = 1 -> Error code in EAX (AL)
                                <1>             ;       stc & EAX = 0 -> Bad command or path name
                                <1>             ;
                                <1>             ; Modified registers: EAX, EBX, ECX, EDX, ESI, EDI
                                <1>
000129C8 66A3[608D0100]         <1>             mov   [SWP_Mode], ax
000129CE A0[6E810100]           <1>             mov   al, [Current_Drv]
000129D3 30E4                   <1>             xor   ah, ah
000129D5 66A3[628D0100]         <1>             mov   [SWP_DRV], ax
                                <1>
                                <1>             ; TRDOS 386 ring 3 (user's page directory)
                                <1>             ; to ring 0 (kernel's page directory)
                                <1>             ; transfer modifications (05/10/2016).
                                <1>
000129DB 55                     <1>             push  ebp
000129DC 89E5                   <1>             mov   ebp, esp
                                <1>
000129DE B980000000             <1>             mov   ecx, 128 ; maximum path length = 128 bytes
000129E3 29CC                   <1>             sub   esp, ecx ; reserve 128 bytes (buffer) on stack
000129E5 89E7                   <1>             mov   edi, esp ; destination address (kernel space)
                                <1>             ; esi = source address (virtual, in user's memory space)
000129E7 E8A0EBFFFF             <1>             call  transfer_from_user_buffer
000129EC 720A                   <1>             jc    short loc_swp_xor_retn
                                <1>
000129EE 89E6                   <1>             mov   esi, esp ; temporary buffer (the path) on stack
                                <1> loc_swp_fchar:
000129F0 8A06                   <1>             mov   al, [esi]
000129F2 3C20                   <1>             cmp   al, 20h
000129F4 7711                   <1>             ja    short loc_swp_parse_path_name
000129F6 740C                   <1>             je    short loc_swp_fchar_next
                                <1>
                                <1> loc_swp_xor_retn:
000129F8 31C0                   <1>             xor   eax, eax
000129FA F9                     <1>             stc
                                <1> loc_swp_retn:
000129FB 89EC                   <1>             mov   esp, ebp
000129FD 5D                     <1>             pop   ebp
                                <1>
                                <1>             ;mov   esi, FindFile_Drv
000129FE BE[508A0100]           <1>             mov   esi, FindFile_Name ; 12/10/2016
00012A03 C3                     <1>             retn
                                <1>
                                <1> loc_swp_fchar_next:
00012A04 46                     <1>             inc   esi
00012A05 EBE9                   <1>             jmp   short loc_swp_fchar
                                <1>
                                <1> loc_swp_parse_path_name:
00012A07 BF[0E8A0100]           <1>             mov   edi, FindFile_Drv
00012A0C E81587FFFF             <1>             call  parse_path_name
00012A11 72E8                   <1>             jc    short loc_swp_retn
                                <1>
                                <1> loc_swp_checkfile_name:
00012A13 803D[608D0100]00       <1>             cmp   byte [SWP_Mode], 0
00012A1A 761E                   <1>             jna   short loc_swp_drv
                                <1>
                                <1>             ; 10/10/2016 (valid file name checking)
00012A1C BE[508A0100]           <1>             mov   esi, FindFile_Name
00012A21 803E20                 <1>             cmp   byte [esi], 20h
00012A24 76D2                   <1>             jna   short loc_swp_xor_retn
                                <1>
                                <1>             ; 16/10/2016
00012A26 C605[5F8D0100]00       <1>             mov   byte [SWP_inv_fname], 0 ; reset
                                <1>             ; esi = file name address (ASCIIZ)
00012A2D E8E168FFFF             <1>             call  check_filename
00012A32 7306                   <1>             jnc   short loc_swp_drv
                                <1>
00012A34 FE05[5F8D0100]         <1>             inc   byte [SWP_inv_fname] ; set
                                <1> loc_swp_drv:
00012A3A 8A35[6E810100]         <1>             mov   dh, [Current_Drv]
                                <1>             ;mov   [RUN_CDRV], dh
```

```
4098                                    <1>
4099 00012A40 8A15[0E8A0100]            <1>             mov     dl, [FindFile_Drv]
4100                                    <1>             ;cmp    dl, dh
4101 00012A46 3A15[6E810100]            <1>             cmp     dl, [Current_Drv]
4102 00012A4C 740D                      <1>             je      short loc_swp_change_directory
4103                                    <1>
4104 00012A4E FE05[638D0100]            <1>             inc     byte [SWP_DRV_chg]
4105 00012A54 E85951FFFF                <1>             call    change_current_drive
4106 00012A59 72A0                      <1>             jc      short loc_swp_retn ; eax = error code
4107                                    <1>             ; eax = 0
4108                                    <1>
4109                                    <1> loc_swp_change_directory:
4110 00012A5B 803D[0F8A0100]21          <1>             cmp     byte [FindFile_Directory], 21h
4111 00012A62 F5                        <1>             cmc
4112 00012A63 7396                      <1>             jnc     short loc_swp_retn
4113                                    <1>
4114 00012A65 FE05[638D0100]            <1>             inc     byte [SWP_DRV_chg]
4115 00012A6B FE05[55390100]            <1>             inc     byte [Restore_CDIR]
4116 00012A71 BE[0F8A0100]              <1>             mov     esi, FindFile_Directory
4117 00012A76 8A25[618D0100]            <1>             mov     ah, [SWP_Mode+1]
4118 00012A7C E88F80FFFF                <1>             call    change_current_directory
4119 00012A81 0F8274FFFFFF              <1>             jc      loc_swp_retn ; eax = error code
4120                                    <1>
4121                                    <1> loc_swp_change_prompt_dir_string:
4122                                    <1>             ; esi = PATH_Array
4123                                    <1>             ; eax = Current Directory First Cluster
4124                                    <1>             ; edi = Logical DOS Drive Description Table
4125 00012A87 E8A97FFFFF                <1>             call    change_prompt_dir_str
4126 00012A8C 29C0                      <1>             sub     eax, eax ; 0
4127 00012A8E E968FFFFFF                <1>             jmp     loc_swp_retn
4128                                    <1>
4129                                    <1> reset_working_path:
4130                                    <1>             ; 06/10/2016 - TRDOS 386 (TRDOS v2.0)
4131                                    <1>             ;
4132                                    <1>             ; TRDOS v1.0 (DIR.ASM, "proc_reset_working_path")
4133                                    <1>             ; 05/02/2011 - 08/02/2011
4134                                    <1>             ;
4135                                    <1>             ; Restores current drive and directory
4136                                    <1>             ;
4137                                    <1>             ; INPUT: none
4138                                    <1>             ; OUTPUT: DL = SWP_DRV, EAX = 0 -> OK
4139                                    <1>             ;
4140                                    <1>             ;    AX = 0 -> ESI = Logical Dos Drv Desc. Table
4141                                    <1>             ;
4142                                    <1>             ;    EAX, EBX, ECX, EDX, ESI, EDI will be changed
4143                                    <1>             ;
4144                                    <1>
4145                                    <1>
4146 00012A93 31C0                      <1>             xor     eax, eax
4147 00012A95 48                        <1>             dec     eax
4148                                    <1>
4149 00012A96 668B15[628D0100]          <1>             mov     dx, [SWP_DRV]
4150 00012A9D 08F6                      <1>             or      dh, dh
4151 00012A9F 742E                      <1>             jz      short loc_rwp_return
4152                                    <1>
4153 00012AA1 3A15[6E810100]            <1>             cmp     dl, [Current_Drv]
4154 00012AA7 7407                      <1>             je      short loc_rwp_restore_cdir
4155                                    <1> loc_rwp_restore_cdrv:
4156 00012AA9 E80451FFFF                <1>             call    change_current_drive
4157 00012AAE EB10                      <1>             jmp     short loc_rwp_restore_ok
4158                                    <1> loc_rwp_restore_cdir:
4159 00012AB0 31DB                      <1>             xor     ebx, ebx
4160 00012AB2 88D7                      <1>             mov     bh, dl
4161 00012AB4 BE00010900                <1>             mov     esi, Logical_DOSDisks
4162 00012AB9 01DE                      <1>             add     esi, ebx
4163                                    <1>
4164 00012ABB E8A951FFFF                <1>             call    restore_current_directory
4165                                    <1>
4166                                    <1> loc_rwp_restore_ok:
4167 00012AC0 668B15[628D0100]          <1>             mov     dx, [SWP_DRV]
4168 00012AC7 31C0                      <1>             xor     eax, eax
4169 00012AC9 66A3[638D0100]            <1>             mov     [SWP_DRV_chg], ax
4170                                    <1> loc_rwp_return:
4171 00012ACF C3                        <1>             retn
4172                                    <1>
4173                                    <1> get_file_name:
4174                                    <1>             ; 15/10/2016 - TRDOS 386 (TRDOS v2.0)
4175                                    <1>             ; Convert file name
4176                                    <1>             ;      from directory entry format
4177                                    <1>             ;      to (8.3) dot file name format
4178                                    <1>             ;
4179                                    <1>             ; TRDOS v1.0 (DIR.ASM, "get_file_name")
4180                                    <1>             ; 2005 - 09/10/2011
4181                                    <1>             ; INPUT:
4182                                    <1>             ;      DS:SI -> Directory Entry Format File Name
4183                                    <1>             ;       ES:DI -> DOS Dot File Name Address
4184                                    <1>             ; OUTPUT:
4185                                    <1>             ;      DS:SI -> DOS Dot File Name Address
4186                                    <1>             ;      ES:DI -> Directory Entry Format File Name
4187                                    <1>             ;
4188                                    <1>             ; TRDOS 386 (15/10/2016)
4189                                    <1>             ; INPUT:
4190                                    <1>             ;      ESI = File name addr in dir entry format
4191                                    <1>             ;      EDI = Dot file name address (destination)
4192                                    <1>             ; OUTPUT:
4193                                    <1>             ;      File name is converted and moved
4194                                    <1>             ;      to destination (as 8.3 dot filename)
4195                                    <1>             ;
4196                                    <1>             ; Modified registers: EAX, ECX
4197                                    <1>
4198                                    <1>              ; 2005 (TRDOS 8086) - 2016 (TRDOS 386)
4199                                    <1>
4200 00012AD0 57                        <1>             push    edi
4201 00012AD1 56                        <1>             push    esi
4202 00012AD2 AC                        <1>             lodsb
```

```
4203 00012AD3 3C20                <1>             cmp    al, 20h
4204 00012AD5 762A                <1>             jna    short pass_gfn_ext
4205 00012AD7 56                  <1>             push   esi
4206 00012AD8 AA                  <1>             stosb
4207 00012AD9 B907000000          <1>             mov    ecx, 7
4208                              <1> loc_gfn_next_char:
4209 00012ADE AC                  <1>             lodsb
4210 00012ADF 3C20                <1>             cmp    al, 20h
4211 00012AE1 7603                <1>             jna    short pass_gfn_fn
4212 00012AE3 AA                  <1>             stosb
4213 00012AE4 E2F8                <1>             loop   loc_gfn_next_char
4214                              <1> pass_gfn_fn:
4215 00012AE6 5E                  <1>             pop    esi
4216 00012AE7 83C607              <1>             add    esi, 7
4217 00012AEA AC                  <1>             lodsb
4218 00012AEB 3C20                <1>             cmp    al, 20h
4219 00012AED 7612                <1>             jna    short pass_gfn_ext
4220 00012AEF B42E                <1>             mov    ah, '.'
4221 00012AF1 86E0                <1>             xchg   ah, al
4222 00012AF3 66AB                <1>             stosw
4223 00012AF5 AC                  <1>             lodsb
4224 00012AF6 3C20                <1>             cmp    al, 20h
4225 00012AF8 7607                <1>             jna    short pass_gfn_ext
4226 00012AFA AA                  <1>             stosb
4227 00012AFB AC                  <1>             lodsb
4228 00012AFC 3C20                <1>             cmp    al, 20h
4229 00012AFE 7601                <1>             jna    short pass_gfn_ext
4230 00012B00 AA                  <1>             stosb
4231                              <1> pass_gfn_ext:
4232 00012B01 30C0                <1>             xor    al, al
4233 00012B03 AA                  <1>             stosb
4234 00012B04 5E                  <1>             pop    esi
4235 00012B05 5F                  <1>             pop    edi
4236 00012B06 C3                  <1>             retn
4237                              <1>
4238                              <1> set_hardware_int_vector:
4239                              <1>             ; 18/03/2017
4240                              <1>             ; 03/03/2017
4241                              <1>             ; 28/02/2017 - TRDOS 386 (TRDOS v2.0)
4242                              <1>             ;
4243                              <1>             ; SET/RESET HARDWARE INTERRUPT GATE
4244                              <1>             ;
4245                              <1>             ; Changes interrupt gate descriptor table
4246                              <1>             ; (without changing default interrupt list)
4247                              <1>             ;
4248                              <1>             ; INPUT:
4249                              <1>             ;      AL = IRQ number (0 to 15)
4250                              <1>             ;      AH > 0 -> set
4251                              <1>             ;      AH = 0 -> reset
4252                              <1>             ;
4253                              <1>             ; Modified registers: eax, ebx, edx, edi
4254                              <1>             ;
4255                              <1>
4256 00012B07 C0E002              <1>             shl    al, 2 ; IRQ number * 4
4257 00012B0A 0FB6D8              <1>             movzx  ebx, al
4258                              <1>
4259 00012B0D 08E4                <1>             or     ah, ah
4260 00012B0F 7508                <1>             jnz    short shintv_1 ; set (for user call service)
4261                              <1>
4262                              <1>             ; 18/03/2017
4263 00012B11 81C3[38410100]      <1>             add    ebx, IRQ_list ; reset to default interrupt list
4264 00012B17 EB06                <1>             jmp    short shintv_2
4265                              <1> shintv_1:
4266 00012B19 81C3[402B0100]      <1>             add    ebx, IRQ_u_list
4267                              <1> shintv_2:
4268 00012B1F 8B13                <1>             mov    edx, [ebx] ; IRQ handler address
4269                              <1>
4270                              <1>             ; 03/03/2017
4271 00012B21 D0E0                <1>             shl    al, 1 ; IRQ number * 8
4272                              <1>             ; 18/03/2017
4273 00012B23 0FB6F8              <1>             movzx  edi, al
4274 00012B26 81C7[C07E0100]      <1>             add    edi, idt + (8*32) ; IRQ 0 offset = idt + 256
4275                              <1>
4276 00012B2C 89D0                <1>             mov    eax, edx ; IRQ handler address
4277 00012B2E BB00000800          <1>             mov    ebx, 80000h
4278                              <1>
4279                              <1>             ;mov   edx, eax
4280 00012B33 66BA008E            <1>             mov    dx, 8E00h
4281 00012B37 6689C3              <1>             mov    bx, ax
4282 00012B3A 89D8                <1>             mov    eax, ebx ; /* selector = 0x0008 = cs */
4283                              <1>                                     ; /* interrupt gate - dpl=0, present */
4284 00012B3C AB                  <1>             stosd  ; selector & offset bits 0-15
4285 00012B3D 8917                <1>             mov    [edi], edx ; attributes & offset bits 16-23
4286                              <1>
4287 00012B3F C3                  <1>             retn
4288                              <1> IRQ_u_list:
4289                              <1>             ; 28/02/2017
4290 00012B40 [50090000]          <1>             dd     timer_int
4291 00012B44 [BE100000]          <1>             dd     kb_int
4292 00012B48 [320B0000]          <1>             dd     irq2
4293 00012B4C [802B0100]          <1>             dd     IRQ_service3
4294 00012B50 [8A2B0100]          <1>             dd     IRQ_service4
4295 00012B54 [942B0100]          <1>             dd     IRQ_service5
4296 00012B58 [CE500000]          <1>             dd     fdc_int
4297 00012B5C [9E2B0100]          <1>             dd     IRQ_service7
4298 00012B60 [BB0A0000]          <1>             dd     rtc_int
4299 00012B64 [A82B0100]          <1>             dd     IRQ_service9
4300 00012B68 [B22B0100]          <1>             dd     IRQ_service10
4301 00012B6C [BC2B0100]          <1>             dd     IRQ_service11
4302 00012B70 [C62B0100]          <1>             dd     IRQ_service12
4303 00012B74 [D02B0100]          <1>             dd     IRQ_service13
4304 00012B78 [2E5A0000]          <1>             dd     hdc1_int
4305 00012B7C [515A0000]          <1>             dd     hdc2_int
4306                              <1>
4307                              <1>             ; 03/03/2017
```

```
                                <1>             ; 27/02/2017
                                <1> IRQ_service3:
00012B80 36C605[D48F0100]03     <1>             mov    byte [ss:IRQnum], 3
00012B88 EB4E                   <1>             jmp    short IRQ_service
                                <1> IRQ_service4:
00012B8A 36C605[D48F0100]04     <1>             mov    byte [ss:IRQnum], 4
00012B92 EB44                   <1>             jmp    short IRQ_service
                                <1> IRQ_service5:
00012B94 36C605[D48F0100]05     <1>             mov    byte [ss:IRQnum], 5
00012B9C EB3A                   <1>             jmp    short IRQ_service
                                <1> IRQ_service7:
00012B9E 36C605[D48F0100]07     <1>             mov    byte [ss:IRQnum], 7
00012BA6 EB30                   <1>             jmp    short IRQ_service
                                <1> IRQ_service9:
00012BA8 36C605[D48F0100]09     <1>             mov    byte [ss:IRQnum], 9
00012BB0 EB26                   <1>             jmp    short IRQ_service
                                <1> IRQ_service10:
00012BB2 36C605[D48F0100]0A     <1>             mov    byte [ss:IRQnum], 10
00012BBA EB1C                   <1>             jmp    short IRQ_service
                                <1> IRQ_service11:
00012BBC 36C605[D48F0100]0B     <1>             mov    byte [ss:IRQnum], 11
00012BC4 EB12                   <1>             jmp    short IRQ_service
                                <1> IRQ_service12:
00012BC6 36C605[D48F0100]0C     <1>             mov    byte [ss:IRQnum], 12
00012BCE EB08                   <1>             jmp    short IRQ_service
                                <1> IRQ_service13:
00012BD0 36C605[D48F0100]0D     <1>             mov    byte [ss:IRQnum], 13
                                <1>             ;jmp   short IRQ_service
                                <1> IRQ_service:
                                <1>             ; 13/06/2017
                                <1>             ; 11/06/2017
                                <1>             ; 10/06/2017
                                <1>             ; 01/03/2017, 04/03/2017
                                <1>             ; 27/02/2017, 28/02/2017
00012BD8 1E                     <1>             push   ds
00012BD9 06                     <1>             push   es
00012BDA 0FA0                   <1>             push   fs
00012BDC 0FA8                   <1>             push   gs
                                <1>
00012BDE 60                     <1>             pushad ; eax,ecx,edx,ebx,esp,ebp,esi,edi
00012BDF 66B91000               <1>             mov     cx, KDATA
00012BE3 8ED9                   <1>             mov     ds, cx
00012BE5 8EC1                   <1>             mov     es, cx
00012BE7 8EE1                   <1>             mov     fs, cx
00012BE9 8EE9                   <1>             mov     gs, cx
                                <1>
00012BEB 0F20D8                 <1>             mov    eax, cr3
00012BEE A3[D08F0100]           <1>             mov    [IRQ_cr3], eax
                                <1>
00012BF3 A1[A8800100]           <1>             mov    eax, [k_page_dir]
00012BF8 0F22D8                 <1>             mov    cr3, eax
                                <1>
00012BFB A0[D48F0100]           <1>             mov    al, [IRQnum]
                                <1>
                                <1>             ;mov   cl, [sysflg]
                                <1>             ;mov   [u.r_mode], cl  ; system (0) or user mode (FFh)
                                <1> IRQsrv_0:
00012C00 0FB6D8                 <1>             movzx  ebx, al
00012C03 8A9B[70400100]         <1>             mov    bl, [ebx+IRQenum] ; IRQ (available) index number + 1
                                <1>             ; 01/03/2017
00012C09 FECB                   <1>             dec    bl  ; IRQ index number, 0 to 8
00012C0B 0F8807010000           <1>             js     IRQsrv_5 ; not available to use here!?
                                <1>             ;
00012C11 80BB[9A8F0100]80       <1>             cmp    byte [ebx+IRQ.method], 80h ; using by a dev or kernel?
00012C18 7205                   <1>             jb     short IRQsrv_1 ; no
                                <1>
                                <1>             ; If the IRQ service is already owned by TRDOS 386 kernel
                                <1>             ;       or a Device driver
                                <1>             ; we need to call 'dev_IRQ_service'
                                <1>
                                <1>             ; IRQ number in AL
00012C1A E81A010000             <1>             call   dev_IRQ_service     ; IRQ service for device drivers
                                <1>             ; IRQ number in AL
                                <1> IRQsrv_1:
                                <1>             ; check user callback service status
                                <1>             ; AL = IRQ number
                                <1>             ; EBX = IRQ (Available) Index number
                                <1>
00012C1F A2[D7030300]           <1>             mov    [u.irqwait], al ; set waiting IRQ flag
                                <1>
00012C24 8A83[888F0100]         <1>             mov    al, [ebx+IRQ.owner]
00012C2A 20C0                   <1>             and    al, al
00012C2C 0F84E6000000           <1>             jz     IRQsrv_5 ; it is not owned by a user/proc
                                <1>
                                <1>             ; 03/03/2017
00012C32 89DA                   <1>             mov    edx, ebx
00012C34 C0E202                 <1>             shl    dl, 2
00012C37 8B92[AC8F0100]         <1>             mov    edx, [edx+IRQ.addr] ; S.R.B. or Callback service addr
                                <1>
00012C3D 8AA3[9A8F0100]         <1>             mov    ah, [ebx+IRQ.method]
00012C43 F6C401                 <1>             test   ah, 1
00012C46 7534                   <1>             jnz    short IRQsrv_4 ; Callback service method
                                <1>
                                <1>             ; Signal Response Byte method
                                <1>             ;mov   edx, [edx+IRQ.addr] ; Signal Response Byte address
                                <1>             ;                          ; (Physical address, non-swappable)
00012C48 80E402                 <1>             and    ah, 2 ; bit 1, (S.R.B.) counter (auto increment) method
00012C4B 8AA3[A38F0100]         <1>             mov    ah, [ebx+IRQ.srb] ; Signal Response Byte value
00012C51 7408                   <1>             jz     short IRQsrv_2 ; fixed S.R.B. value
                                <1>             ; counter method (auto increment)
00012C53 FEC4                   <1>             inc    ah
00012C55 88A3[A38F0100]         <1>             mov    [ebx+IRQ.srb], ah ; next (count) number
                                <1> IRQsrv_2:
00012C5B 8822                   <1>             mov    [edx], ah ; put S.R.B. val to the user's S.R.B. addr
00012C5D C605[D7030300]00       <1>             mov    byte [u.irqwait], 0 ; clear waiting IRQ flag
```

```
                                <1>
00012C64 3A05[B3030300]         <1>                 cmp    al, [u.uno]
00012C6A 0F84A8000000           <1>                 je     IRQsrv_5 ; the owner is current user/process
                                <1> IRQsrv_3:
                                <1>                 ; the owner is not current user/process
                                <1>                 ; AL = process number
00012C70 B202                   <1>                 mov    dl, 2 ; priority, 2 = event (high)
00012C72 E837FAFFFF             <1>                 call   set_run_sequence
                                <1>
                                <1>                 ; [u.irqwait] = waiting IRQ number for callback service
                                <1>
00012C77 E99C000000             <1>                 jmp    IRQsrv_5
                                <1> IRQsrv_4:
00012C7C 3A05[B3030300]         <1>                 cmp    al, [u.uno]  ; is the owner is current user/process?
00012C82 75EC                   <1>                 jne    short IRQsrv_3 ; no !
                                <1>
                                <1>                 ; Check if an IRQ callback service already in progress
00012C84 803D[D8030300]00       <1>                 cmp    byte [u.r_lock], 0
00012C8B 0F8787000000           <1>                 ja     IRQsrv_5 ; nothing to do !
                                <1>                                  ; (we need to complete prev callback)
00012C91 803D[D4030300]00       <1>                 cmp    byte [u.t_lock], 0
00012C98 777E                   <1>                 ja     short IRQsrv_5 ; nothing to do !
                                <1>                                  ; (we need to complete timer callback)
                                <1>
                                <1>                 ; 04/03/2017
00012C9A C605[D7030300]00       <1>                 mov    byte [u.irqwait], 0 ; reset/clear waiting IRQ flag
                                <1>
00012CA1 FE05[D8030300]         <1>                 inc    byte [u.r_lock] ; 'IRQ callback service in progress' flag
                                <1>
00012CA7 8A0D[5B030300]         <1>                 mov    cl, [sysflg]   ; (system call) mode flag (kernel/user)
00012CAD 880D[D9030300]         <1>                 mov    [u.r_mode], cl ; system mode (0) or user mode (FFh)
                                <1>
                                <1>                 ;
00012CB3 8B2D[44800100]         <1>                 mov    ebp, [tss.esp0] ; kernel stack address (for ring 0)
00012CB9 83ED14                 <1>                 sub    ebp, 20             ; eip, cs, eflags, esp, ss
00012CBC 892D[5C030300]         <1>                 mov    [u.sp], ebp
00012CC2 8925[60030300]         <1>                 mov    [u.usp], esp
                                <1>
                                <1>                 ;or    word [ebp+8], 200h ; 22/01/2017, force enabling interrupts
                                <1>
00012CC8 8B44241C               <1>                 mov    eax, [esp+28] ; pushed eax
00012CCC A3[64030300]           <1>                 mov    [u.r0], eax
                                <1>
00012CD1 E820E7FFFF             <1>                 call   wswap ; save user's registers & status
                                <1>
                                <1>                 ; software int is in ring 0 but IRQ handler must return to ring 3
                                <1>                 ; so, ring 3 return address and stack registers
                                <1>                 ; (eip, cs, eflags, esp, ss)
                                <1>                 ; must be copied to IRQ handler return
                                <1>                 ; eip will be replaced by callback service routine address
                                <1>
00012CD6 C605[5B030300]FF       <1>                 mov    byte [sysflg], 0FFh ; user mode
                                <1>
                                <1>                 ; system mode (system call)
                                <1>                 ;mov   ebp, [u.sp] ; EIP (u), CS (UCODE), EFLAGS (u),
                                <1>                                    ; ESP (u), SS (UDATA)
                                <1>
00012CDD 8B4510                 <1>                 mov    eax, [ebp+16]; SS (UDATA)
00012CE0 89E6                   <1>                 mov    esi, esp
00012CE2 50                     <1>                 push   eax
00012CE3 50                     <1>                 push   eax
00012CE4 89E7                   <1>                 mov    edi, esp
00012CE6 893D[60030300]         <1>                 mov    [u.usp], edi
00012CEC B908000000             <1>                 mov    ecx, ((ESPACE/4) - 4) ; except DS, ES, FS, GS
00012CF1 F3A5                   <1>                 rep    movsd
00012CF3 B104                   <1>                 mov    cl, 4
00012CF5 F3AB                   <1>                 rep    stosd
00012CF7 893D[5C030300]         <1>                 mov    [u.sp], edi
00012CFD 89EE                   <1>                 mov    esi, ebp
00012CFF B105                   <1>                 mov    cl, 5 ; EIP (u), CS (UCODE), EFLAGS (u), ESP (u), SS (UDATA)
00012D01 F3A5                   <1>                 rep    movsd
                                <1>                 ;
                                <1>
00012D03 8B0D[B8030300]         <1>                 mov    ecx, [u.pgdir]
00012D09 890D[D08F0100]         <1>                 mov    [IRQ_cr3], ecx
                                <1>
                                <1> set_IRQ_callback_addr:
                                <1>                 ;
                                <1>                 ; This routine sets return address
                                <1>                 ; to start of user's interrupt
                                <1>                 ; service (callback) address
                                <1>                 ;
                                <1>                 ; INPUT:
                                <1>                 ;      EDX = callback routine/service address
                                <1>                 ;            (virtual, not physical address!)
                                <1>                 ;      [u.sp] = kernel stack, points to
                                <1>                 ;               user's EIP,CS,EFLAGS,ESP,SS
                                <1>                 ;               registers.
                                <1>                 ; OUTPUT:
                                <1>                 ;      EIP (user) = callback (service) address
                                <1>                 ;      CS (user) = UCODE
                                <1>                 ;      EFLAGS (user) = flags before callback
                                <1>                 ;        ESP (user) = ESP-4 (user, before callback)
                                <1>                 ;      [ESP](user) = EIP (user) before callback
                                <1>                 ;
                                <1>                 ; Note: If CPU was in user mode while entering
                                <1>                 ;      the timer interrupt service routine,
                                <1>                 ;      'IRET' will get return to callback routine
                                <1>                 ;      immediately. If CPU was in system/kernel mode
                                <1>                 ;      'iret' will get return to system call and
                                <1>                 ;      then, callback routine will be return address
                                <1>                 ;      from system call. (User's callback/service code
                                <1>                 ;      will be able to return to normal return address
                                <1>                 ;      via a 'sysrele' system call at the end.)
                                <1>                 ;
```

```
                                <1>             ; Note: User's IRQ callback service code must be ended
                                <1>             ;      with a 'sysrele' system call !
                                <1>             ;
                                <1>             ;      For example:
                                <1>             ;
                                <1>             ;      audio_IRQ_callback:
                                <1>             ;          ...
                                <1>             ;          <load DMA buffer with audio data>
                                <1>             ;          ...
                                <1>             ;          mov eax, 39 ; 'sysrele'
                                <1>             ;          int 40h ; TRDOS 386 system call (interrupt)
                                <1>             ;
                                <1>
                                <1>             ;mov   edx, [edx+IRQ.addr] ; Callback service address
                                <1>             ;                          ; (Virtual address)
                                <1>
00012D0F 8B2D[5C030300]         <1>             mov    ebp, [u.sp]; kernel's stack, points to EIP (user)
00012D15 895500                 <1>             mov    [ebp], edx
                                <1> IRQsrv_5:
                                <1>             ; EOI & return
                                <1>             ; 01/08/2020
                                <1>             ; 11/06/2017
                                <1>             ; 10/06/2017
                                <1>             ;mov   al, [IRQnum]
00012D18 B020                   <1>             mov    al, 20h ; 01/08/2020
00012D1A FA                     <1>             cli
                                <1>             ;cmp   al, 7
00012D1B 803D[D48F0100]07       <1>             cmp    byte [IRQnum], 7 ; 01/08/2020
00012D22 7602                   <1>             jna    short IRQsrv_6
                                <1>             ;
                                <1>             ;;mov  al, EOI      ; end of interrupt
                                <1>             ;mov   al, 20h ; 01/08/2020
                                <1>             ;cli         ; disable interrupts till stack cleared
                                <1>             ;out   INTB00, al ; For controll2 #2
00012D24 E6A0                   <1>             out    0A0h, al
                                <1> IRQsrv_6:
                                <1>             ;mov   byte [IRQnum], 0 ; reset
                                <1>             ;;mov  al, EOI      ; end of interrupt
                                <1>             ;mov   al, 20h ; 01/08/2020
                                <1>             ;cli         ; disable interrupts till stack cleared
                                <1>             ;out   INTA00, al ; end of interrupt to 8259 - 1
00012D26 E620                   <1>             out    20h, al
                                <1> IRQsrv_7:
                                <1>             ;; 13/06/2017
                                <1>             ;or    word [ebp+8], 200h ; force enabling interrupts
                                <1>             ;
00012D28 8B0D[D08F0100]         <1>             mov    ecx, [IRQ_cr3]      ; previous content of cr3 register
00012D2E 0F22D9                 <1>             mov    cr3, ecx     ; restore cr3 register content
                                <1>             ;
00012D31 61                     <1>             popad ; edi,esi,ebp,(icrement esp by 4),ebx,edx,ecx,eax
                                <1>             ;
00012D32 0FA9                   <1>             pop    gs
00012D34 0FA1                   <1>             pop    fs
00012D36 07                     <1>             pop    es
00012D37 1F                     <1>             pop    ds
                                <1>             ;
00012D38 CF                     <1>             iretd  ; return from interrupt
                                <1>
                                <1> ; 17/04/2021
                                <1> ; ('get_device_number' procedure is disabled as temporary)
                                <1>
                                <1> ;get_device_number:
                                <1> ;           ; 08/10/2016
                                <1> ;           ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
                                <1> ;           ;
                                <1> ;           ; This procedure compares name of requested
                                <1> ;           ; device with kernel device names and
                                <1> ;           ; installable device names. If names match,
                                <1> ;           ; the relevant device index (entry) number
                                <1> ;           ; will be returned the caller (sysopen)
                                <1> ;           ; for the requested device.
                                <1> ;           ;
                                <1> ;           ; NOTE: Installable device drivers must
                                <1> ;           ; be loaded before using 'sysopen'
                                <1> ;           ; (opendev) system call.
                                <1> ;           ;
                                <1> ;           ; INPUT:
                                <1> ;           ;    ESI = device name address (ASCIIZ)
                                <1> ;           ;         (in kernel's memory space)
                                <1> ;           ;    max name length = 8 without '/dev/')
                                <1> ;           ;    Device name will be capitalized
                                <1> ;           ;    and if there is, '/dev/' will be
                                <1> ;           ;    removed from name before comparising)
                                <1> ;           ;
                                <1> ;           ; OUTPUT:
                                <1> ;           ;    cf = 0 ->
                                <1> ;           ;      EAX (AL) = device entry/index number
                                <1> ;           ;    cf = 1 -> device not found (installed)
                                <1> ;           ;              or invalid device name
                                <1> ;           ;              (AL=0)
                                <1> ;           ;    device_name = device name address (asciiz)
                                <1> ;           ;      ;
                                <1> ;           ; Modified registers: EAX, EBX, ESI, EDI
                                <1> ;
                                <1> ;           mov    edi, device_name
                                <1> ;           call   lodsb_capitalize
                                <1> ;           mov    ah, al
                                <1> ;           cmp    al, '/'
                                <1> ;           jne    short gdn_1
                                <1> ;           mov    edi, device_name
                                <1> ;           call   lodsb_capitalize
                                <1> ;gdn_0:
                                <1> ;           and    al, al ; 0 ?
                                <1> ;           jz     short gdn_err ; null name after '/'
                                <1> ;gdn_1:
```

```
4623                    <1> ;           cmp     al, 'D'
4624                    <1> ;           jne     short gdn_2
4625                    <1> ;           call    lodsb_capitalize
4626                    <1> ;           cmp     al, 'E'
4627                    <1> ;           jne     short gdn_2
4628                    <1> ;           call    lodsb_capitalize
4629                    <1> ;           cmp     al, 'V'
4630                    <1> ;           jne     short gdn_2
4631                    <1> ;           lodsb
4632                    <1> ;           cmp     al, '/'
4633                    <1> ;           je      short gdn_4
4634                    <1> ;gdn_2:
4635                    <1> ;           cmp     ah, '/'
4636                    <1> ;           jne     short gdn_5
4637                    <1> ;gdn_err:
4638                    <1> ;           ; invalid device name or device not found
4639                    <1> ;           xor     eax, eax ; 0
4640                    <1> ;           stc
4641                    <1> ;           retn
4642                    <1> ;gdn_3:
4643                    <1> ;           cmp     al, '/'
4644                    <1> ;           jne     short gdn_5
4645                    <1> ;gdn_4:
4646                    <1> ;           mov     edi, device_name
4647                    <1> ;           jmp     short gdn_6
4648                    <1> ;gdn_5:
4649                    <1> ;           cmp     al, 0
4650                    <1> ;           je      short gdn_7
4651                    <1> ;gdn_6:
4652                    <1> ;           call    lodsb_capitalize
4653                    <1> ;           cmp     edi, device_name + 8
4654                    <1> ;           jb      short gdn_3
4655                    <1> ;           cmp     al, 0
4656                    <1> ;           jne     short gdn_err
4657                    <1> ;           cmp     edi, device_name + 1
4658                    <1> ;           jna     short gdn_err ; null name after '/'
4659                    <1> ;gdn_7:
4660                    <1> ;           stosb
4661                    <1> ;           ; zero padding ("NAME",0,0,0,0)
4662                    <1> ;           cmp     edi, device_name + 8
4663                    <1> ;           jb      short gdn_7
4664                    <1> ;gdn_8:
4665                    <1> ;           ; search for kernel device names
4666                    <1> ;           mov     esi, device_name
4667                    <1> ;           mov     edi, KDEV_NAME
4668                    <1> ;           xor     eax, eax
4669                    <1> ;gdn_9:
4670                    <1> ;           cmpsd
4671                    <1> ;           jne     short gdn_10
4672                    <1> ;           cmpsd
4673                    <1> ;           jne     short gdn_11
4674                    <1> ;           jmp     short gdn_17 ; match
4675                    <1> ;gdn_10:
4676                    <1> ;           cmpsd  ; add esi, 4 & add edi, 4
4677                    <1> ;gdn_11:
4678                    <1> ;           mov     esi, device_name
4679                    <1> ;           inc     al
4680                    <1> ;           cmp     al, NumOfKernelDevNames
4681                    <1> ;           jb      short gdn_9
4682                    <1> ;gdn_12:
4683                    <1> ;           ; search for installable device names
4684                    <1> ;           ; esi = offset device_name
4685                    <1> ;           mov     edi, IDEV_NAME
4686                    <1> ;           sub     al, al ; 0
4687                    <1> ;gdn_13:
4688                    <1> ;           cmpsd
4689                    <1> ;           jne     short gdn_14
4690                    <1> ;           cmpsd
4691                    <1> ;           jne     short gdn_15
4692                    <1> ;           jmp     short gdn_19 ; match
4693                    <1> ;gdn_14:
4694                    <1> ;           cmpsd  ; add esi, 4 & add edi, 4
4695                    <1> ;gdn_15:
4696                    <1> ;           mov     esi, device_name
4697                    <1> ;           inc     al
4698                    <1> ;           cmp     al, NumOfInstallableDevices
4699                    <1> ;           jb      short gdn_13
4700                    <1> ;
4701                    <1> ;gdn_16:    ; error: invalid device name (not found) !
4702                    <1> ;           xor     al, al
4703                    <1> ;           stc
4704                    <1> ;           retn
4705                    <1> ;
4706                    <1> ;gdn_17:    ; name match (with one of kernel device names)
4707                    <1> ;           ;
4708                    <1> ;           ; convert KDEV_NAME index to
4709                    <1> ;           ; KDEV_NUMBER index
4710                    <1> ;           ; (different names are used for same devices)
4711                    <1> ;           ; (example: "COM1" & "TTY8" = device number 18)
4712                    <1> ;           mov     ebx, eax ; < 256
4713                    <1> ;           mov     al, [KDEV_NUMBER+ebx]
4714                    <1> ;
4715                    <1> ;           ; check if empty dev entry in the list
4716                    <1> ;           cmp     byte [DEV_OPENMODE+eax], 0
4717                    <1> ;           ja      short gdn_18 ; it must be already set
4718                    <1> ;
4719                    <1> ;           ; (re)set device name and access flags
4720                    <1> ;           ; (remain open work will be easy after that)
4721                    <1> ;           ; (NOTE: here, data will be copied to bss section)
4722                    <1> ;           mov     bl, al
4723                    <1> ;           sub     edi, 8 ; kernel device name address (data)
4724                    <1> ;           shl     bx, 2
4725                    <1> ;           mov     [DEV_NAME_PTR+ebx], edi ; (all) device names
4726                    <1> ;           mov     bl, [KDEV_ACCESS+eax] ; kernel dev list (data)
4727                    <1> ;           mov     [DEV_ACCESS+eax], bl ; (all) device list (bss)
```

```
4728                    <1> ;gdn_18:
4729                    <1> ;           inc    al ; 1 to NumOfKernelDevNames (<=7Fh)
4730                    <1> ;           ; eax = device index/entry number
4731                    <1> ;           retn
4732                    <1> ;
4733                    <1> ;gdn_19:    ; name match (with one of installable device names)
4734                    <1> ;           ;
4735                    <1> ;           ; al = 0 to NumOfInstallableDevices - 1 (<=7Fh)
4736                    <1> ;
4737                    <1> ;           mov    ebx, eax
4738                    <1> ;           add    bl, NumOfKernelDevices    ; < NUMOFDEVICES
4739                    <1> ;
4740                    <1> ;           ; check if empty dev entry in the list
4741                    <1> ;           cmp    byte [DEV_OPENMODE+ebx], 0
4742                    <1> ;           ja     short gdn_20 ; it must be already set
4743                    <1> ;
4744                    <1> ;           ; (re)set device name and access flags
4745                    <1> ;           ; (remain open work will be easy after that)
4746                    <1> ;           sub    edi, 8 ; installable device name address
4747                    <1> ;           shl    bx, 2 ;*4
4748                    <1> ;           mov    [DEV_NAME_PTR+ebx], edi ; (all) device names
4749                    <1> ;           shr    bx, 2
4750                    <1> ;           mov    al, [IDEV_FLAGS+eax] ; installable dev list
4751                    <1> ;           mov    [DEV_ACCESS+ebx], al ; (all) device list
4752                    <1> ;gdn_20:
4753                    <1> ;           mov    al, bl
4754                    <1> ;           ; eax = device index/entry number ; < NUMOFDEVICES
4755                    <1> ;           retn
4756                    <1>
4757                    <1> ;lodsb_capitalize:
4758                    <1> ;     ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
4759                    <1> ;     ; INPUT -> [esi] = character
4760                    <1> ;     ;          edi = destination
4761                    <1> ;     ; OUTPUT -> AL contains capitalized character
4762                    <1> ;     ;         esi = esi+1
4763                    <1> ;     ;         edi = edi+1
4764                    <1> ;     ;
4765                    <1> ;     lodsb
4766                    <1> ;     cmp    al, 61h
4767                    <1> ;     jb     short lodsb_cap_retn
4768                    <1> ;            cmp    al, 7Ah
4769                    <1> ;            ja     short lodsb_cap_retn
4770                    <1> ;            and    al, 0DFh
4771                    <1> ;lodsb_cap_retn:
4772                    <1> ;     stosb
4773                    <1> ;     retn
4774                    <1>
4775                    <1> ; 17/04/2021
4776                    <1> ; ('device_open' procedure is disabled as temporary)
4777                    <1>
4778                    <1> ;device_open:
4779                    <1> ;     ; 08/10/2016 - TRDOS 386 (TRDOS v2.0)
4780                    <1> ;     ; Complete device opening work for sysopen (device)
4781                    <1> ;     ;
4782                    <1> ;     ; INPUT ->
4783                    <1> ;     ;      EAX = Device Number (AL)
4784                    <1> ;     ;        CL = Open mode (1 = read, 2 = write)
4785                    <1> ;     ;       CH = Device access byte (bit 0 = 0)
4786                    <1> ;     ; OUTPUT ->
4787                    <1> ;     ;      EAX = Device Number
4788                    <1> ;     ;      CF = 0 -> device has been opened
4789                    <1> ;     ;      CF = 1 -> device could not be opened
4790                    <1> ;     ;
4791                    <1> ;     ;  Modified registers: ebx, (edx, ecx, esi, edi, ebp)
4792                    <1> ;     ;
4793                    <1> ;
4794                    <1> ;     mov    ebx, eax
4795                    <1> ;     shl    bx, 2 ; *4
4796                    <1> ;
4797                    <1> ;     test   ch, 80h ; bit 7, installable device driver flag
4798                    <1> ;     jz     short d_open_2 ; Kernel device
4799                    <1> ;     ; installable device
4800                    <1> ;d_open_1:
4801                    <1> ;       jmp  dword [ebx+IDEV_OADDR-4]
4802                    <1> ;d_open_2:
4803                    <1> ;     jmp    dword [ebx+KDEV_OADDR-4]
4804                    <1>
4805                    <1> ; 17/04/2021
4806                    <1> ; ('device_close' procedure is disabled as temporary)
4807                    <1>
4808                    <1> ;device_close:
4809                    <1> ;     ; 08/10/2016 - TRDOS 386 (TRDOS v2.0)
4810                    <1> ;     ; Complete device closing work for sysclose (device)
4811                    <1> ;     ;
4812                    <1> ;     ; INPUT ->
4813                    <1> ;     ;      EAX = Device Number (AL)
4814                    <1> ;     ;        CL = Open mode (1 = read, 2 = write)
4815                    <1> ;     ;       CH = Device access byte (bit 0 = 0)
4816                    <1> ;     ; OUTPUT ->
4817                    <1> ;     ;      EAX = Device Number
4818                    <1> ;     ;      CF = 0 -> device has been closed
4819                    <1> ;     ;      CF = 1 -> device could not be closed
4820                    <1> ;     ;
4821                    <1> ;     ; Modified registers: ebx, (edx, ecx, esi, edi, ebp)
4822                    <1> ;     ;
4823                    <1> ;
4824                    <1> ;     mov    ebx, eax
4825                    <1> ;     shl    bx, 2 ; *4
4826                    <1> ;
4827                    <1> ;     test   ch, 80h ; bit 7, installable device driver flag
4828                    <1> ;     jz     short d_close_2 ; Kernel device
4829                    <1> ;     ; installable device
4830                    <1> ;d_close_1:
4831                    <1> ;       jmp  dword [ebx+IDEV_CADDR-4]
4832                    <1> ;d_close_2:
```

```
4833                                    <1> ;       jmp     dword [ebx+KDEV_CADDR-4]
4834                                    <1>
4835                                    <1> ;rnull:
4836                                    <1> ;       ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
4837                                    <1> ;       ; read null (read from null device)
4838                                    <1> ;       retn
4839                                    <1>
4840                                    <1> ;wnull:
4841                                    <1> ;       ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
4842                                    <1> ;       ; write null (write to null device)
4843                                    <1> ;       retn
4844                                    <1>
4845                                    <1> dev_IRQ_service:
4846                                    <1>         ; 12/05/2017
4847                                    <1>         ; 13/04/2017
4848                                    <1>         ; 27/02/2017 - TRDOS 386 (TRDOS v2.0)
4849                                    <1>         ; INPUT ->
4850                                    <1>         ;       AL = IRQ Number (0 to 15)
4851                                    <1>         ;
4852 00012D39 53                       <1>         push    ebx
4853 00012D3A 0FB6D8                   <1>         movzx   ebx, al
4854 00012D3D C0E302                   <1>         shl     bl, 2 ; * 4
4855 00012D40 8B9B[488F0100]           <1>         mov     ebx, [ebx+DEV_INT_HNDLR]
4856 00012D46 21DB                     <1>         and     ebx, ebx
4857 00012D48 7404                     <1>           jz    short dIRQ_s_retn
4858 00012D4A 50                       <1>         push    eax
4859                                    <1>
4860 00012D4B FFD3                     <1>         call    ebx
4861                                    <1>
4862 00012D4D 58                       <1>         pop     eax
4863                                    <1> dIRQ_s_retn:
4864 00012D4E 5B                       <1>         pop     ebx
4865 00012D4F C3                       <1>         retn
4866                                    <1>
4867                                    <1> set_dev_IRQ_service:
4868                                    <1>         ; 13/04/2017 - TRDOS 386 (TRDOS v2.0)
4869                                    <1>         ;
4870                                    <1>         ; Set Device Interrupt Service
4871                                    <1>         ;
4872                                    <1>         ; INPUT ->
4873                                    <1>         ;       AL = IRQ Number
4874                                    <1>         ;       EBX = Hardware Interrupt Service Address
4875                                    <1>         ;
4876                                    <1>         ; Note: There is not a validation check here
4877                                    <1>         ;       because this procedure is called by
4878                                    <1>         ;       TRDOS 386 kernel !
4879                                    <1>         ;         (Even if a device driver does not exist
4880                                    <1>         ;       this setting may be used by sysaudio
4881                                    <1>         ;       and other system calls for hardware
4882                                    <1>         ;       components which use IRQ method for I/O.)
4883                                    <1>         ;
4884                                    <1>         ;push  esi
4885 00012D50 0FB6F0                   <1>         movzx  esi, al
4886 00012D53 66C1E602                 <1>         shl    si, 2 ; * 4
4887 00012D57 899E[488F0100]           <1>         mov    [esi+DEV_INT_HNDLR], ebx
4888                                    <1>         ;pop   esi
4889 00012D5D C3                       <1>         retn
4890                                    <1>
4891                                    <1>
4892                                    <1> sysaudio: ; AUDIO FUNCTIONS
4893                                    <1>         ; 12/02/2021 (TRDOS 386 v2.0.3)
4894                                    <1>         ; 28/07/2020
4895                                    <1>         ; 27/07/2020
4896                                    <1>         ; 10/10/2017
4897                                    <1>         ; 22/06/2017
4898                                    <1>         ; 28/05/2017, 04/06/2017, 05/06/2017, 10/06/2017
4899                                    <1>         ; 01/05/2017, 12/05/2017, 15/05/2017, 20/05/2017
4900                                    <1>         ; 21/04/2017, 22/04/2017, 23/04/2017, 24/04/2017
4901                                    <1>         ; 10/04/2017, 13/04/2017, 14/04/2017, 16/04/2017
4902                                    <1>         ; 03/04/2017 (VIA VT8237R)
4903                                    <1>         ; 01/04/2016 (trdosk6.s -> tdosk8.s)
4904                                    <1>         ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
4905                                    <1>         ;
4906                                    <1>         ; Inputs:
4907                                    <1>         ;
4908                                    <1>         ;       BH = 0 -> Beep (PC Speaker)
4909                                    <1>         ;            BL = Duration Counter (1 for 1/64 second)
4910                                    <1>         ;            CX = Frequency Divisor (1193180/Frequency)
4911                                    <1>         ;               (1331 for 886 Hz)
4912                                    <1>         ;
4913                                    <1>         ;       01/04/2017
4914                                    <1>         ;
4915                                    <1>         ;       BH = 1 -> DETECT (& ENABLE) AUDIO DEVICE
4916                                    <1>         ;            BL = 0 : PC SPEAKER
4917                                    <1>         ;                 1 : SOUND BLASTER 16
4918                                    <1>         ;                 2 : INTEL AC'97
4919                                    <1>         ;                 3 : VIA VT8237R (VT8233)
4920                                    <1>         ;                 4 : INTEL HDA
4921                                    <1>         ;              5-FEh : unknown/invalid
4922                                    <1>         ;                ; 04/06/2017
4923                                    <1>         ;               FFh : Get current audio device id
4924                                    <1>         ;
4925                                    <1>         ;       BH = 2 -> ALLOCATE AUDIO BUFFER (for user)
4926                                    <1>         ;             ECX = Audio Buffer Size (must be equal to
4927                                    <1>         ;                   the half of DMA buffer size)
4928                                    <1>         ;             EDX = Virtual Address of the buffer
4929                                    <1>         ;                   (This is not DMA buffer!)
4930                                    <1>         ;
4931                                    <1>         ;       BH = 3 -> INITIALIZE AUDIO DEVICE
4932                                    <1>         ;            BL = 0,2 -> for Signal Response Byte
4933                                    <1>         ;             CL = Signal Response Byte Value (fixed)
4934                                    <1>         ;                         if BL = 0
4935                                    <1>         ;                  auto increment of S.R.B. value
4936                                    <1>         ;                         if BL = 2
4937                                    <1>         ;               EDX = Signal Response (Return) Byte Address
```

```
4938                    <1>     ;
4939                    <1>     ;           BL = 1 for CallBack Method
4940                    <1>     ;             EDX = CallBack Service Address (Virtual)
4941                    <1>     ;
4942                    <1>     ;           BL > 2 -> invalid function
4943                    <1>     ;
4944                    <1>     ;          (Audio buffer must be allocated before
4945                    <1>     ;           initialization.)
4946                    <1>     ;
4947                    <1>     ;      BH = 4 -> START TO PLAY
4948                    <1>     ;           BL = Mode
4949                    <1>     ;               Bit 0 = mono/stereo (1 = stereo)
4950                    <1>     ;               Bit 1 = 8 bit / 16 bit (1 = 16 bit)
4951                    <1>     ;           CX = Sampling Rate (Hz)
4952                    <1>     ;
4953                    <1>     ;      BH = 5 -> PAUSE
4954                    <1>     ;           BL = Any
4955                    <1>     ;
4956                    <1>     ;      BH = 6 -> CONTINUE TO PLAY
4957                    <1>     ;           BL = Any
4958                    <1>     ;
4959                    <1>     ;      BH = 7 -> STOP
4960                    <1>     ;           BL = Any
4961                    <1>     ;
4962                    <1>     ;      BH = 8 -> RESET
4963                    <1>     ;           BL = Any
4964                    <1>     ;
4965                    <1>     ;      BH = 9 -> CANCEL (CALLBACK or S.R.B. SERVICE)
4966                    <1>     ;           BL = Any
4967                    <1>     ;
4968                    <1>     ;      BH = 10 -> DEALLOCATE AUDIO BUFFER (for user)
4969                    <1>     ;           BL = Any
4970                    <1>     ;
4971                    <1>     ;      BH = 11 -> SET VOLUME LEVEL
4972                    <1>     ;           BL: (Bit 0 to 6)
4973                    <1>     ;               0 = Master (Playback, Lineout) volume
4974                    <1>     ;           CL = Left Channel Volume
4975                    <1>     ;           CH = Right Channel Volume
4976                    <1>     ;
4977                    <1>     ;           Note: If BL >= 80h (Bit 7 of BL is set),
4978                    <1>     ;           volume level will be set for next playing
4979                    <1>     ;           (actual volume level will not be changed
4980                    <1>     ;           immediately)
4981                    <1>     ;
4982                    <1>     ;      BH = 12 -> DISABLE AUDIO DEVICE
4983                    <1>     ;           (reset audio device and unlink dma buffer)
4984                    <1>     ;           BL = Any
4985                    <1>     ;
4986                    <1>     ;      12/05/2017
4987                    <1>     ;      BH = 13 -> MAP DMA BUFFER TO USER
4988                    <1>     ;          (for direct access to system's dma buffer)
4989                    <1>     ;
4990                    <1>     ;           ECX = map size in bytes
4991                    <1>     ;               (will be rounded up to page borders)
4992                    <1>     ;           EDX = Virtual Address of the buffer
4993                    <1>     ;               (Will be rounded up to page borders)
4994                    <1>     ;
4995                    <1>     ;      05/06/2017
4996                    <1>     ;      04/06/2017
4997                    <1>     ;      BH = 14 -> GET AUDIO DEVICE INFO
4998                    <1>     ;           BL: 0 = Audio Controller Info
4999                    <1>     ;             > 0 = Invalid for now!
5000                    <1>     ;
5001                    <1>     ;      22/06/2017
5002                    <1>     ;      BH = 15 -> GET CURRENT SOUND DATA (for graphics)
5003                    <1>     ;           BL: 0 -> PCM OUT data
5004                    <1>     ;             > 0 -> Invalid for now!
5005                    <1>     ;           ECX = 0 -> Get DMA Buffer Pointer
5006                    <1>     ;              EDX = Not Used
5007                    <1>     ;           ECX > 0 -> Byte count for buffer (EDX)
5008                    <1>     ;               EDX = Buffer Address (Virtual)
5009                    <1>     ;
5010                    <1>     ;      10/10/2017
5011                    <1>     ;      BH = 16 -> UPDATE DMA BUFFER DATA
5012                    <1>     ;                (by using the Audio Buffer content)
5013                    <1>     ;           BL = 0 : Update dma half buffer in sequence
5014                    <1>     ;                    (automatic destination)
5015                    <1>     ;               1 : Update 1st half of the dma buffer
5016                    <1>     ;               2 : Update 2nd half of the dma buffer
5017                    <1>     ;               3-FEh: Invalid!
5018                    <1>     ;               FFh = Get current flag value
5019                    <1>     ;                    (Half buffer number -1)
5020                    <1>     ;
5021                    <1>     ;
5022                    <1>     ; Outputs:
5023                    <1>     ;
5024                    <1>     ;      For BH = 0 -> Beep
5025                    <1>     ;          None
5026                    <1>     ;
5027                    <1>     ;      01/04/2017
5028                    <1>     ;
5029                    <1>     ;      For BH = 1 -> DETECT (& ENABLE) AUDIO DEVICE
5030                    <1>     ;          AH = 0 : PC SPEAKER
5031                    <1>     ;               1 : SOUND BLASTER 16
5032                    <1>     ;               2 : INTEL AC'97
5033                    <1>     ;               3 : VIA VT8237R (VT8233)
5034                    <1>     ;               4 : INTEL HDA
5035                    <1>     ;             5-FFh : unknown/invalid
5036                    <1>     ;          AL = mode status
5037                    <1>     ;              bit 0 = mono /stereo (1 = stereo)
5038                    <1>     ;              bit 1 = 8 bit / 16 bit ( 1 = 16 bit)
5039                    <1>     ;          04/06/2017
5040                    <1>     ;          EBX = PCI DEVICE/VENDOR ID (if >0)
5041                    <1>     ;              (BX = VENDOR ID)
5042                    <1>     ;          (if CF = 1 -> Error code in EAX)
```

```
                                <1>       ;
                                <1>       ;      For BH = 2 -> ALLOCATE AUDIO BUFFER (for user)
                                <1>       ;          EAX = Physical Address of the buffer
                                <1>       ;          (if CF = 1 -> Error code in EAX)
                                <1>       ;
                                <1>       ;      For BH = 3 -> INITIALIZE AUDIO DEVICE
                                <1>       ;          (if CF = 1 -> Error code in EAX)
                                <1>       ;
                                <1>       ;      For BH = 4 -> START TO PLAY
                                <1>       ;          none (if CF = 1 -> Error code in EAX)
                                <1>       ;
                                <1>       ;      For BH = 5 -> PAUSE
                                <1>       ;          none (if CF = 1 -> Error code in EAX)
                                <1>       ;
                                <1>       ;      For BH = 6 -> CONTINUE TO PLAY
                                <1>       ;          none (if CF = 1 -> Error code in EAX)
                                <1>       ;
                                <1>       ;      For BH = 7 -> STOP
                                <1>       ;          none (if CF = 1 -> Error code in EAX)
                                <1>       ;
                                <1>       ;      For BH = 8 -> RESET
                                <1>       ;          none (if CF = 1 -> Error code in EAX)
                                <1>       ;
                                <1>       ;      For BH = 9 -> CANCEL (CALLBACK or S.R.B. SERVICE)
                                <1>       ;          none (if CF = 1 -> Error code in EAX)
                                <1>       ;
                                <1>       ;      For BH = 10 -> DEALLOCATE AUDIO BUFFER (for user)
                                <1>       ;          none (if CF = 1 -> Error code in EAX)
                                <1>       ;
                                <1>       ;      For BH = 11 -> SET VOLUME LEVEL
                                <1>       ;          none (if CF = 1 -> Error code in EAX)
                                <1>       ;
                                <1>       ;      For BH = 12 -> DISABLE AUDIO DEVICE
                                <1>       ;          none (if CF = 1 -> Error code in EAX)
                                <1>       ;
                                <1>       ;      12/05/2017
                                <1>       ;      For BH = 13 -> MAP DMA BUFFER TO USER
                                <1>       ;          EAX = Physical Address of the buffer
                                <1>       ;          (if CF = 1 -> Error code in EAX)
                                <1>       ;
                                <1>       ;      04/06/2017
                                <1>       ;      For BH = 14 -> GET AUDIO DEVICE INFO
                                <1>       ;      (for BL = 0) ; 05/06/2017
                                <1>       ;          EAX = IRQ Number in AL
                                <1>       ;               Audio Device Number in AH
                                <1>       ;          EBX = DEV/VENDOR ID
                                <1>       ;              (DDDDDDDDDDDDDDDDVVVVVVVVVVVVVVVV)
                                <1>       ;          ECX = BUS/DEV/FN
                                <1>       ;              (00000000BBBBBBBBDDDDDFFF00000000)
                                <1>       ;          EDX = NABMBAR/NAMBAR (for AC97)
                                <1>       ;               (Low word, DX = NAMBAR address)
                                <1>       ;          EDX = Base IO Addr (DX) for SB16 & VT8233
                                <1>       ;          (if CF = 1 -> Error code in EAX)
                                <1>       ;                        (ERR_DEV_NOT_RDY = 15)
                                <1>       ;
                                <1>       ;      22/06/2017
                                <1>       ;      For BH = 15 -> GET CURRENT SOUND DATA
                                <1>       ;                     (for graphics)
                                <1>       ;      (for BL = 0)
                                <1>       ;       If ECX input is 0
                                <1>       ;          EAX = DMA Buffer Current Position (Offset)
                                <1>       ;       If ECX input >  0
                                <1>       ;          EAX = Actual transfer count
                                <1>       ;          (Sound samples will be copied from
                                <1>       ;           Current DMA Buffer Position to EDX
                                <1>       ;           virtual address as EAX bytes.)
                                <1>       ;        ((If CF = 1 ->  Error code in EAX))
                                <1>       ;
                                <1>       ;
                                <1>       ;      10/10/2017
                                <1>       ;      For BH = 16 -> UPDATE DMA BUFFER DATA
                                <1>       ;          EAX = 0, if the updated (or current)
                                <1>       ;                    half buffer is DMA half buffer 1
                                <1>       ;          EAX = 1, if the updated (or current)
                                <1>       ;                    half buffer is DMA half buffer 2
                                <1>       ;          (If CF = 1 -> Error code in EAX)
                                <1>       ;
                                <1>
00012D5E 80FF11                 <1>       cmp    bh, AUDIO1L/4
00012D61 0F8369A6FFFF           <1>       jnb    sysret
                                <1>
00012D67 C0E702                 <1>       shl    bh, 2 ; *4
00012D6A 0FB6F7                 <1>       movzx  esi, bh
                                <1>
                                <1>       ; 22/04/2017
00012D6D 31C0                   <1>       xor    eax, eax
00012D6F A3[64030300]           <1>       mov    [u.r0], eax  ; 0
                                <1>
00012D74 FF96[7F2D0100]         <1>       call   dword [esi+AUDIO1]
                                <1>       ;jc    error
00012D7A E951A6FFFF             <1>       jmp    sysret
                                <1>
00012D7F [44230000]             <1> AUDIO1:      dd     _beep ; 12/02/2021
                                <1>       ;dd    beep ; FUNCTION = 0 (bl = Duration Counter
                                <1>                   ;                     cx = Frequency Divisor
00012D83 [C32D0100]             <1>       dd     soundc_detect
00012D87 [5F2E0100]             <1>       dd     sound_alloc
00012D8B [1D2F0100]             <1>       dd     soundc_init
00012D8F [D5300100]             <1>       dd     sound_play
00012D93 [71310100]             <1>       dd     sound_pause
00012D97 [9B310100]             <1>       dd     sound_continue
00012D9B [C5310100]             <1>       dd     sound_stop
00012D9F [EE310100]             <1>       dd     soundc_reset
00012DA3 [1F320100]             <1>       dd     soundc_cancel
00012DA7 [45320100]             <1>       dd     sound_dalloc
```

```
5148 00012DAB [70320100]          <1>       dd     sound_volume
5149 00012DAF [C2320100]          <1>       dd     soundc_disable
5150 00012DB3 [34330100]          <1>       dd     sound_dma_map
5151 00012DB7 [A3330100]          <1>       dd     soundc_info
5152 00012DBB [02340100]          <1>       dd     sound_data
5153 00012DBF [AF340100]          <1>       dd     sound_update
5154                              <1>
5155                              <1> AUDIO1L     EQU    $ - AUDIO1
5156                              <1>
5157                              <1> soundc_detect:
5158                              <1>       ; FUNCTION = 1
5159                              <1>       ; bl = Audio device type number
5160                              <1>       ; (0= pc speaker, 1 = sound blaster 16, 2 = intel ac97
5161                              <1>       ;  3= via vt823x, 4 = intel HDA, 0FFh= any)
5162                              <1>
5163                              <1>       ; 04/06/2017
5164 00012DC3 8A25[D98F0100]      <1>       mov    ah, [audio_device]
5165 00012DC9 80FBFF              <1>       cmp    bl, 0FFh ; get current audio device id
5166 00012DCC 7408                <1>       je     short sysaudio0
5167                              <1>
5168 00012DCE 20E4                <1>       and    ah, ah
5169 00012DD0 741E                <1>       jz     short soundc_get_dev
5170                              <1>
5171 00012DD2 38DC                <1>       cmp    ah, bl
5172 00012DD4 7567                <1>       jne    short soundc_dev_err
5173                              <1>
5174                              <1> sysaudio0:
5175 00012DD6 A0[DA8F0100]        <1>       mov    al, [audio_mode]
5176                              <1> sysaudio1:
5177 00012DDB A3[64030300]        <1>       mov    [u.r0], eax
5178 00012DE0 8B1D[E48F0100]      <1>       mov    ebx, [audio_vendor] ; (DEVICE/VENDOR ID)
5179 00012DE6 8B2D[60030300]      <1>       mov    ebp, [u.usp]
5180 00012DEC 895D10              <1>       mov    [ebp+16], ebx  ; ebx
5181 00012DEF C3                  <1>       retn
5182                              <1>
5183                              <1> soundc_get_dev:
5184                              <1>       ; 28/05/2017
5185                              <1>       ; 03/04/2017, 24/04/2017
5186 00012DF0 C605[D88F0100]00    <1>       mov    byte [audio_pci], 0
5187 00012DF7 80FB03              <1>       cmp    bl, 3 ; VIA VT8233 (VT8237R) Audio Controller & AC97 Codec
5188                              <1>       ;jne   short soundc_get_dev_sb
5189                              <1>       ; 28/05/2017
5190 00012DFA 7220                <1>       jb     short soundc_get_dev_sb
5191 00012DFC 773F                <1>       ja     short soundc_dev_err  ; temporary (28/05/2017)
5192                              <1>       ;
5193 00012DFE E852160000          <1>       call   DetectVT8233
5194 00012E03 7238                <1>       jc     short soundc_dev_err
5195                              <1>       ; eax = 0
5196                              <1>
5197                              <1>       ;mov   ebx, [audio_vendor]
5198                              <1>       ; ebx = DEVICE/VENDOR ID
5199                              <1>       ;      DDDDDDDDDDDDDDDDVVVVVVVVVVVVVVVV
5200                              <1>
5201 00012E05 B003                <1>       mov    al, 3  ; VIA VT8237R (VT3233) Audio Controller
5202 00012E07 88C4                <1>       mov    ah, al
5203                              <1>
5204                              <1> soundc_get_pci_dev_ok: ; 28/05/2017
5205 00012E09 FE05[D88F0100]      <1>       inc    byte [audio_pci] ; = 1
5206                              <1> soundc_get_dev_ok:
5207                              <1>
5208                              <1> soundc_get_dev_sb16_ok:
5209 00012E0F A2[D98F0100]        <1>       mov    [audio_device], al
5210 00012E14 8825[DA8F0100]      <1>       mov    [audio_mode], ah ; stereo (bit0), 16 bit (bit1) capability
5211 00012E1A EBBF                <1>       jmp    short sysaudio1
5212                              <1>
5213                              <1> soundc_get_dev_sb:
5214                              <1>       ; 24/04/2017
5215 00012E1C 80FB01              <1>       cmp    bl, 1 ; Sound Blaster 16
5216 00012E1F 750E                <1>       jne    short soundc_get_dev_ich ; 28/05/2017
5217                              <1>       ;
5218 00012E21 E8531B0000          <1>       call   DetectSB
5219 00012E26 7215                <1>       jc     short soundc_dev_err
5220 00012E28 B801030000          <1>       mov    eax, 0301h ; Sound Blaster 16
5221 00012E2D EBE0                <1>       jmp    short soundc_get_dev_sb16_ok
5222                              <1>
5223                              <1> soundc_get_dev_ich:
5224                              <1>       ; 28/05/2017
5225                              <1>       ;cmp   bl, 2 ; Intel AC'97 Audio Controller (ICH)
5226                              <1>       ;jne   short soundc_dev_err ; Temporary  (28/05/2017)
5227                              <1>       ;                           ; (Here will be modified just after
5228                              <1>       ;                           ; new sound card code will be ready!)
5229 00012E2F E814160000          <1>       call   DetectICH
5230 00012E34 7207                <1>       jc     short soundc_dev_err
5231                              <1>       ;
5232 00012E36 B802030000          <1>       mov    eax, 0302h ; AC'97 (ICH)
5233 00012E3B EBCC                <1>       jmp    short soundc_get_pci_dev_ok
5234                              <1>
5235                              <1> soundc_dev_err:
5236 00012E3D B80F000000          <1>       mov    eax, ERR_DEV_NOT_RDY ; Device not ready !
5237 00012E42 EB0C                <1>       jmp    short sysaudio_err
5238                              <1>
5239                              <1> sound_buff_error:
5240 00012E44 B82E000000          <1>       mov    eax, ERR_BUFFER ; Buffer error !
5241 00012E49 EB05                <1>       jmp    short sysaudio_err
5242                              <1>
5243                              <1> soundc_respond_err:
5244                              <1>       ; ERR_TIME_OUT ; 'time out !' error
5245 00012E4B B819000000          <1>       mov    eax, ERR_DEV_NOT_RESP ; 'device not responding !' error
5246                              <1> sysaudio_err:
5247 00012E50 A3[64030300]        <1>       mov    [u.r0], eax
5248 00012E55 A3[C8030300]        <1>       mov    [u.error], eax
5249 00012E5A E951A5FFFF          <1>       jmp    error
5250                              <1>
5251                              <1> sound_alloc:
5252                              <1>       ; FUNCTION =  2
```

```
                            <1>       ; ecx = audio buffer size (in bytes)
                            <1>       ; edx = audio buffer address (virtual)
                            <1>       ; 27/07/2020
                            <1>       ; 28/05/2017
                            <1>       ; 01/05/2017, 15/05/2017
                            <1>       ; 21/04/2017, 24/04/2017
00012E5F 803D[D88F0100]00   <1>       cmp   byte [audio_pci], 0
00012E66 7708               <1>       ja    short snd_alloc_0
                            <1>       ; Max. 64KB DMA buffer !!!
00012E68 81F900800000       <1>       cmp   ecx, 32768
00012E6E 77D4               <1>       ja    short sound_buff_error
                            <1> snd_alloc_0:
                            <1>       ; 15/05/2017
00012E70 81F900100000       <1>       cmp   ecx, 4096 ; PAGE_SIZE
00012E76 72CC               <1>       jb    short sound_buff_error
                            <1>       ;
00012E78 A1[EC8F0100]       <1>       mov   eax, [audio_buffer] ; audio buffer address (current)
00012E7D 09C0               <1>       or    eax, eax
00012E7F 7445               <1>       jz    short snd_alloc_2
                            <1>       ; audio buffer exists !
00012E81 8A1D[B3030300]     <1>       mov   bl, [u.uno]
00012E87 3A1D[01900100]     <1>       cmp   bl, [audio_user]
00012E8D 0F85FC000000       <1>       jne   sndc_owner_error ; not owner !
00012E93 39D0               <1>       cmp   eax, edx ; same virtual buffer address ?
00012E95 7508               <1>       jne   short snd_alloc_1
00012E97 3B0D[F48F0100]     <1>       cmp   ecx, [audio_buff_size]
00012E9D 746C               <1>       je    short snd_alloc_3 ; Nothing to do !
                            <1>                             ; Buffer has been set already!
                            <1> snd_alloc_1:
00012E9F 51                 <1>       push  ecx
00012EA0 52                 <1>       push  edx
00012EA1 89C3               <1>       mov   ebx, eax ; audio buffer address (current)
00012EA3 8B0D[F48F0100]     <1>       mov   ecx, [audio_buff_size]
00012EA9 E83434FFFF         <1>       call  deallocate_user_pages
00012EAE 5A                 <1>       pop   edx
00012EAF 59                 <1>       pop   ecx
00012EB0 31C0               <1>       xor   eax, eax ; 0
00012EB2 A3[EC8F0100]       <1>       mov   [audio_buffer], eax  ; 0
00012EB7 A3[F08F0100]       <1>       mov   [audio_p_buffer], eax  ; 0
00012EBC A3[F48F0100]       <1>       mov   [audio_buff_size], eax
00012EC1 A2[01900100]       <1>       mov   [audio_user], al ; 0
                            <1> snd_alloc_2:
00012EC6 89D3               <1>       mov   ebx, edx
                            <1>       ; 01/05/2017
00012EC8 BA00F0FFFF         <1>       mov   edx, ~PAGE_OFF ; truncating page offsets
                            <1>                            ; for aligning to page borders
                            <1>       ;and  eax, edx
00012ECD 21D3               <1>       and   ebx, edx
00012ECF 21D1               <1>       and   ecx, edx
                            <1>       ; 15/05/2017
                            <1>       ; EAX = Beginning address (physical)
                            <1>       ; EAX = 0 -> Allocate mem block from the 1st proper aperture
                            <1>       ; ECX = Number of bytes to be allocated
00012ED1 E8C130FFFF         <1>       call  allocate_memory_block
00012ED6 0F8268FFFFFF       <1>       jc    sound_buff_error
                            <1>       ; EAX = Physical address of the allocated memory block
                            <1>       ; ECX = Allocated bytes (as truncated to page border)
                            <1>       ; EBX = Virtual address (as truncated to page border)
00012EDC 50                 <1>       push  eax
00012EDD 53                 <1>       push  ebx
00012EDE 51                 <1>       push  ecx
00012EDF E8E634FFFF         <1>       call  allocate_user_pages
00012EE4 59                 <1>       pop   ecx
00012EE5 5B                 <1>       pop   ebx
00012EE6 58                 <1>       pop   eax
00012EE7 722A               <1>       jc    short snd_alloc_4  ; insufficient memory, buff error
                            <1>       ; eax = physical address of the user's audio buffer
                            <1>       ; ebx = virtual address of the user's audio buffer
                            <1>       ; ecx = buffer size (in bytes)
00012EE9 A3[F08F0100]       <1>       mov   [audio_p_buffer], eax
00012EEE 891D[EC8F0100]     <1>       mov   [audio_buffer], ebx
00012EF4 890D[F48F0100]     <1>       mov   [audio_buff_size], ecx
00012EFA 8A15[B3030300]     <1>       mov   dl, [u.uno]
00012F00 8815[01900100]     <1>       mov   [audio_user], dl
00012F06 A3[64030300]       <1>       mov   [u.r0], eax
                            <1> snd_alloc_3:
                            <1>       ; 27/07/2020
00012F0B C605[00900100]00   <1>       mov   byte [audio_flag], 0 ; clear dma half buffer flag
                            <1>       ;
00012F12 C3                 <1>       retn
                            <1> snd_alloc_4:
                            <1>       ; 15/05/2017
                            <1>       ; EAX = Beginning address (physical)
                            <1>       ; ECX = Number of bytes to be deallocated
00012F13 E88C32FFFF         <1>       call  deallocate_memory_block
00012F18 E927FFFFFF         <1>       jmp   sound_buff_error  ; insufficient memory, buff error
                            <1>
                            <1> soundc_init:
                            <1>       ; FUNCTION = 3
                            <1>       ; bl = method (0= s.r.b., 1= callback, 2= auto incr s.r.b.)
                            <1>       ; cl = signal response byte (initial or fixed) value
                            <1>       ; edx = signal response byte or callback address
                            <1>       ; 27/07/2020
                            <1>       ; 28/05/2017
                            <1>       ; 12/05/2017, 20/05/2017
                            <1>       ; 22/04/2017, 23/04/2017, 24/04/2017
                            <1>       ; 13/04/2017, 14/04/2017, 16/04/2017, 21/04/2017
                            <1>       ; 03/04/2017, 10/04/2017
                            <1>
00012F1D A0[D98F0100]       <1>       mov   al, [audio_device]
00012F22 20C0               <1>       and   al, al
00012F24 7549               <1>       jnz   short sndc_init_6
                            <1>       ;
00012F26 C605[D88F0100]00   <1>       mov   byte [audio_pci], 0
00012F2D 52                 <1>       push  edx
```

```
5358 00012F2E 53                <1>       push   ebx
5359 00012F2F 51                <1>       push   ecx
5360 00012F30 E8441A0000        <1>       call   DetectSB
5361 00012F35 7213              <1>       jc     short sndc_init_8
5362 00012F37 66B80103          <1>       mov    ax, 0301h ; Sound Blaster 16
5363 00012F3B EB1E              <1>       jmp    short sndc_init_7
5364                            <1>
5365                            <1> sndc_init_11:
5366                            <1>       ; 28/05/2017
5367 00012F3D E806150000        <1>       call   DetectICH ; Detect AC'97 (ICH) Audio Controller
5368 00012F42 7217              <1>       jc     short sndc_init_7
5369 00012F44 66B80203          <1>       mov    ax, 0302h ; Intel AC'97 Audio Device
5370 00012F48 EB0B              <1>       jmp    short sndc_init_12 ; (PCI device)
5371                            <1>
5372                            <1> sndc_init_8:
5373 00012F4A E806150000        <1>       call   DetectVT8233
5374                            <1>       ;jc    short sndc_init_7
5375 00012F4F 72EC              <1>       jc     sndc_init_11 ; 28/05/2017
5376                            <1>       ; eax = 0
5377 00012F51 B003              <1>       mov    al, 3  ; VIA VT8237R (VT3233) Audio Controller
5378 00012F53 88C4              <1>       mov    ah, al
5379                            <1>
5380                            <1> sndc_init_12:
5381 00012F55 FE05[D88F0100]    <1>       inc    byte [audio_pci] ; = 1
5382                            <1> sndc_init_7:
5383 00012F5B 59                <1>       pop    ecx
5384 00012F5C 5B                <1>       pop    ebx
5385 00012F5D 5A                <1>       pop    edx
5386 00012F5E 0F82D9FEFFFF      <1>       jc     soundc_dev_err
5387                            <1>       ;
5388 00012F64 A2[D98F0100]      <1>       mov    [audio_device], al
5389 00012F69 8825[DA8F0100]    <1>       mov    [audio_mode], ah ; stereo (bit0), 16 bit (bit1) capability
5390                            <1>
5391                            <1> sndc_init_6:
5392 00012F6F 833D[EC8F0100]00  <1>       cmp    dword [audio_buffer], 0
5393 00012F76 0F86C8FEFFFF      <1>       jna    sound_buff_error
5394                            <1>
5395 00012F7C A0[B3030300]      <1>       mov    al, [u.uno]
5396 00012F81 8A25[01900100]    <1>       mov    ah, [audio_user]
5397 00012F87 08E4              <1>       or     ah, ah
5398 00012F89 7418              <1>       jz     short sndc_init0
5399 00012F8B 38E0              <1>       cmp    al, ah
5400 00012F8D 7419              <1>       je     short sndc_init1
5401                            <1>
5402                            <1> sndc_owner_error:
5403 00012F8F B80B000000        <1>       mov    eax, ERR_NOT_OWNER ; 'permission denied !' error
5404                            <1> sndc_perm_error:
5405 00012F94 A3[64030300]      <1>       mov    [u.r0], eax
5406 00012F99 A3[C8030300]      <1>       mov    [u.error], eax
5407 00012F9E E90DA4FFFF        <1>       jmp    error
5408                            <1> sndc_init0:
5409 00012FA3 A2[01900100]      <1>       mov    [audio_user], al
5410                            <1> sndc_init1:
5411 00012FA8 8915[04900100]    <1>       mov    [audio_cb_addr], edx
5412 00012FAE 881D[02900100]    <1>       mov    [audio_cb_mode], bl
5413 00012FB4 880D[03900100]    <1>       mov    [audio_srb], cl
5414                            <1>
5415                            <1>       ; 27/07/2020
5416                            <1>       ;mov  byte [audio_flag], 0  ; clear dma half buffer flag
5417                            <1>
5418                            <1>       ; 24/04/2017
5419 00012FBA 803D[D98F0100]03  <1>       cmp    byte [audio_device], 3 ; VT8233 (VT8237R)
5420 00012FC1 7438              <1>       je     short sndc_init_9
5421                            <1>       ;ja    short soundc_respond_err ; temporary (28/05/2017)
5422 00012FC3 803D[D98F0100]01  <1>       cmp    byte [audio_device], 1 ; SB 16
5423 00012FCA 7510              <1>       jne    short sndc_init_13
5424 00012FCC BB[9E4B0100]      <1>       mov    ebx, sb16_int_handler
5425                            <1>       ; Note: 'SbInit' is at 'Start to Play' stage
5426                            <1>       ; 20/05/2017
5427 00012FD1 66C705[0E900100]08- <1>     mov    word [audio_master_volume], 0808h ; 2/8
5427 00012FD9 08                <1>
5428 00012FDA EB3F              <1>       jmp    short sndc_init_10
5429                            <1> sndc_init_13:
5430                            <1>       ; 28/05/2017
5431 00012FDC 803D[D98F0100]02  <1>       cmp    byte [audio_device], 2 ; AC 97 (ICH)
5432 00012FE3 0F8562FEFFFF      <1>       jne    soundc_respond_err ; temporary (28/05/2017)
5433                            <1>
5434 00012FE9 E8051D0000        <1>       call   ac97_codec_config
5435 00012FEE 0F8257FEFFFF      <1>       jc     soundc_respond_err ; codec error !
5436                            <1>
5437 00012FF4 BB[DA4E0100]      <1>       mov    ebx, ac97_int_handler
5438 00012FF9 EB20              <1>       jmp    short sndc_init_10
5439                            <1>
5440                            <1> sndc_init_9:
5441                            <1>       ;call  reset_codec
5442                            <1>       ;; eax = 1
5443                            <1>       ;call  codec_io_w16 ; w32
5444 00012FFB E8CC150000        <1>       call   init_codec ; 28/05/2017
5445 00013000 0F8245FEFFFF      <1>       jc     soundc_respond_err ; codec error !
5446                            <1>
5447 00013006 E8F3170000        <1>       call   channel_reset
5448                            <1>
5449                            <1>       ; setup the Codec (actually mixer registers)
5450 0001300B E813170000        <1>       call   codec_config  ; unmute codec, set rates.
5451 00013010 0F8235FEFFFF      <1>       jc     soundc_respond_err ; codec error !
5452                            <1>
5453 00013016 BB[90470100]      <1>       mov    ebx, vt8233_int_handler
5454                            <1> sndc_init_10:
5455                            <1>       ; 13/04/2017
5456 0001301B A0[DB8F0100]      <1>       mov    al, [audio_intr] ; IRQ number
5457 00013020 E82BFDFFFF        <1>       call   set_dev_IRQ_service
5458                            <1>
5459                            <1>       ; SETUP (audio) INTERRUPT CALLBACK SERVICE
5460 00013025 8A1D[DB8F0100]    <1>       mov    bl, [audio_intr] ; IRQ number
5461 0001302B 8A3D[02900100]    <1>       mov    bh, [audio_cb_mode]
```

```
5462 00013031 FEC7                <1>       inc    bh  ; 1 = Signal Response Byte method (fixed value)
5463                              <1>                   ; 2 = Callback service method
5464                              <1>                   ; 3 = Auto Increment S.R.B. method
5465 00013033 8A0D[03900100]      <1>       mov    cl, [audio_srb]
5466 00013039 8B15[04900100]      <1>       mov    edx, [audio_cb_addr]
5467 0001303F A0[01900100]        <1>       mov    al, [audio_user]
5468                              <1>       ; 14/04/2017
5469 00013044 E8E1040000          <1>       call   set_irq_callback_service
5470                              <1>       ; 16/04/2017
5471 00013049 A3[64030300]        <1>       mov    [u.r0], eax
5472                              <1>       ;jnc   sysret
5473 0001304E 7316                <1>       jnc    short sndc_init2 ; 21/04/2017
5474                              <1>       ;
5475 00013050 A3[C8030300]        <1>       mov    dword [u.error], eax
5476                              <1>
5477 00013055 A0[DB8F0100]        <1>       mov    al, [audio_intr] ; IRQ number
5478 0001305A 31DB                <1>       xor    ebx, ebx ; reset IRQ handler address
5479 0001305C E8EFFCFFFF          <1>       call   set_dev_IRQ_service
5480                              <1>
5481 00013061 E94AA3FFFF          <1>       jmp    error
5482                              <1>
5483                              <1> sndc_init2:
5484                              <1>       ; 21/04/2017
5485 00013066 8B0D[F48F0100]      <1>       mov    ecx, [audio_buff_size] ; audio buffer size
5486 0001306C D1E1                <1>       shl    ecx, 1 ; *2
5487 0001306E A1[F88F0100]        <1>       mov    eax, [audio_dma_buff]
5488 00013073 21C0                <1>       and    eax, eax
5489 00013075 7415                <1>       jz     short sndc_init3
5490                              <1>
5491 00013077 8B15[FC8F0100]      <1>       mov    edx, [audio_dmabuff_size] ; dma buffer size
5492 0001307D 39D1                <1>       cmp    ecx, edx
5493 0001307F 744D                <1>       je     short sndc_init5
5494                              <1>
5495 00013081 87CA                <1>       xchg   ecx, edx
5496 00013083 E81C31FFFF          <1>       call   deallocate_memory_block
5497 00013088 87D1                <1>       xchg   edx, ecx
5498 0001308A 31C0                <1>       xor    eax, eax
5499                              <1> sndc_init3:
5500                              <1>       ; 12/05/2017
5501 0001308C 803D[D98F0100]01    <1>       cmp    byte [audio_device], 1 ; SB 16
5502 00013093 7515                <1>       jne    short sndc_init4
5503 00013095 C705[F88F0100]-     <1>       mov    dword [audio_dma_buff], sb16_dma_buffer
5503 0001309B [00000200]          <1>
5504 0001309F C705[FC8F0100]0000- <1>       mov    dword [audio_dmabuff_size], 65536
5504 000130A7 0100                <1>
5505                              <1>       ;xor   eax, eax
5506                              <1>       ;mov   [u.r0], eax ; 0 = no error, successful
5507 000130A9 C3                  <1>       retn
5508                              <1>
5509                              <1> sndc_init4:
5510                              <1>       ; EAX = Beginning address (physical)
5511                              <1>       ; EAX = 0 -> Allocate mem block from the 1st proper aperture
5512                              <1>       ; ECX = Number of bytes to be allocated (>0)
5513 000130AA E8E82EFFFF          <1>       call   allocate_memory_block
5514 000130AF 0F828FFDFFFF        <1>       jc     sound_buff_error
5515                              <1>
5516                              <1>       ; set dma buffer address and size parameters
5517 000130B5 A3[F88F0100]        <1>       mov    [audio_dma_buff], eax ; dma buffer address
5518 000130BA 890D[FC8F0100]      <1>       mov    [audio_dmabuff_size], ecx ; dma buffer size
5519                              <1> ;     ; EAX = Beginning (physical) addr of the allocated mem block
5520                              <1> ;     ; ECX = Num of allocated bytes (rounded up to page borders)
5521                              <1> ;     cmp    byte [audio_pci], 0 ; AC97 audio controller ?
5522                              <1> ;     ja     short sndc_init4
5523                              <1> ;
5524                              <1> ;     ; Sound Blaster 16 uses classic DMA
5525                              <1> ;     mov    edx, eax
5526                              <1> ;     add    edx, ecx
5527                              <1> ;     cmp    edx, 1000000h ; 1st 16 MB
5528                              <1> ;     jna    short sndc_init4
5529                              <1> ;
5530                              <1> ;     ; error !
5531                              <1> ;     ; restore Memory Allocation Table Content
5532                              <1> ;     ; EAX = Beginning address (physical)
5533                              <1> ;     ; ECX = Number of bytes to be deallocated
5534                              <1> ;     call   deallocate_memory_block
5535                              <1> ;     ; reset dma buffer address and size parameters
5536                              <1> ;     xor    eax, eax ; 0
5537                              <1> ;     mov    [audio_dma_buff], eax ; 0
5538                              <1> ;     mov    [audio_dmabuff_size], ecx ; 0
5539                              <1> ;     jmp    sound_buff_error
5540                              <1> ;
5541                              <1> ;sndc_init4:
5542 000130C0 803D[D98F0100]03    <1>       cmp    byte [audio_device], 3
5543                              <1>       ;jne   short sndc_init5
5544 000130C7 7506                <1>       jne    short sndc_init14 ; 28/05/2017
5545 000130C9 E871170000          <1>       call   set_vt8233_bdl
5546                              <1> sndc_init5:
5547                              <1>       ;sub   eax, eax ; 0
5548                              <1>       ;mov   [u.r0], eax ; 0 = no error, successful
5549 000130CE C3                  <1>       retn
5550                              <1> sndc_init14:
5551 000130CF E8381D0000          <1>       call   set_ac97_bdl
5552                              <1>       ;jmp   short sndc_init5
5553 000130D4 C3                  <1>       retn
5554                              <1>
5555                              <1> sound_play:
5556                              <1>       ; FUNCTION = 4
5557                              <1>       ; bl = Mode
5558                              <1>       ;      bit 0 = mono/stereo (1 = stereo)
5559                              <1>       ;      bit 1 = 8 bit / 16 bit (1 = 16 bit)
5560                              <1>       ; cx = Sampling Rate (Hz)
5561                              <1>
5562                              <1>       ; 13/06/2017
5563                              <1>       ; Note: Even if Mode bits are not 11b,
5564                              <1>       ;      AC'97 Audio Controller (&Codec)
```

```
                                <1>       ;      will play audio samples as 16 bit, stereo
                                <1>       ;      samples.
                                <1>       ;      (Program must fill the audio buffer
                                <1>       ;      as required; 8 bit samples must be converted
                                <1>       ;      to 16 bit samples and mono samples must be
                                <1>       ;      converted to stereo samples...)
                                <1>       ;
                                <1>       ; 28/07/2020
                                <1>       ; 27/07/2020
                                <1>       ; 28/05/2017
                                <1>       ; 15/05/2017, 20/05/2017
                                <1>       ; 21/04/2017, 24/04/2017
                                <1>       ; ... device check at first
000130D5 A0[D98F0100]           <1>       mov    al, [audio_device]
000130DA 08C0                   <1>       or     al, al ; 0 ; pc speaker or invalid
000130DC 0F8473F2FEFF           <1>       jz     beeper_gfx ; 'video.s' ; temporary !
                                <1> ;     cmp    al, 3 ; VIA VT 8237R (vt8233)
                                <1> ;     je     short snd_play_1
                                <1> ;     cmp    al, 1 ; SB 16
                                <1> ;     jne    soundc_dev_err ; temporary !
                                <1> ;snd_play_0:
                                <1>       ; ... buffer & (buffer) owner check at second
000130E2 833D[EC8F0100]00       <1>       cmp    dword [audio_buffer], 0
000130E9 0F8655FDFFFF           <1>       jna    sound_buff_error
000130EF A0[B3030300]           <1>       mov    al, [u.uno]
000130F4 3A05[01900100]         <1>       cmp    al, [audio_user]
000130FA 0F858FFEFFFF           <1>       jne    sndc_owner_error
                                <1>
00013100 66890D[0A900100]       <1>       mov    [audio_freq], cx ; sample frequency (Hertz)
00013107 88D8                   <1>       mov    al, bl
00013109 2401                   <1>       and    al, 1 ; mono/stereo (1= stereo)
0001310B FEC0                   <1>       inc    al ; channels
0001310D A2[09900100]           <1>       mov    [audio_stmo], al ; sound channels (1 or 2)
00013112 B008                   <1>       mov    al, 8
00013114 F6C302                 <1>       test   bl, 2 ; bits per sample (1= 16 bit)
00013117 7402                   <1>       jz     short snd_play_bps
00013119 D0E0                   <1>       shl    al, 1
                                <1> snd_play_bps:
0001311B A2[08900100]           <1>       mov    [audio_bps], al
                                <1>
                                <1>       ; Transfer ring 3 (user's) audio buffer content to dma buffer
00013120 8B3D[F88F0100]         <1>       mov    edi, [audio_dma_buff] ; dma buffer (ring 0)
00013126 09FF                   <1>       or     edi, edi
00013128 0F8416FDFFFF           <1>       jz     sound_buff_error
                                <1>
                                <1>       ; 27/07/2020
                                <1>
0001312E 8B35[F08F0100]         <1>       mov    esi, [audio_p_buffer] ; physical address (ring 3)
                                <1>       ;mov   ecx, [audio_buff_size] ; 15/05/2017
00013134 8B0D[FC8F0100]         <1>       mov    ecx, [audio_dmabuff_size] ; 27/07/2020
                                <1>       ;or    ecx, ecx
                                <1>       ;jz    sound_buff_error
                                <1>       ; 28/07/2020
0001313A D1E9                   <1>       shr    ecx, 1  ; dma half buffer size
                                <1>
0001313C 8035[00900100]01       <1>       xor    byte [audio_flag], 1 ;  0 -> 1, 1 -> 0
00013143 7502                   <1>       jnz    short snd_play_0 ; [audio_flag] = 1
                                <1>                             ; fill dma half buffer 1
                                <1>       ; [audio_flag] = 0
                                <1>
                                <1>       ; fill dma half buffer 2
00013145 01CF                   <1>       add    edi, ecx
                                <1>
                                <1> snd_play_0:
                                <1>       ;rep   movsb
00013147 C1E902                 <1>       shr    ecx, 2  ; convert byte count to dword count
0001314A F3A5                   <1>       rep    movsd
                                <1>
                                <1>       ; here, if [audio_flag] = 0, interrupt handler will update
                                <1>                             ; dma half buffer 2
                                <1>                             ; (user's audio buffer data will be
                                <1>                             ; copied into dma half buffer 2)
                                <1>       ;; 20/05/2017
                                <1>       ;mov   byte [audio_flag], 1 ; next half (on next time)
                                <1>
                                <1>       ; 24/04/2017
0001314C A0[D98F0100]           <1>       mov    al, [audio_device]
00013151 3C03                   <1>       cmp    al, 3 ; VT8233 (VT8237R)
00013153 7410                   <1>       je     short snd_play_1
00013155 3C01                   <1>       cmp    al, 1 ; Sound Blaster 16
00013157 7512                   <1>       jne    short snd_play_2  ; 28/05/2017
00013159 E8E9180000             <1>       call   SbInit_play
0001315E 0F82E7FCFFFF           <1>       jc     soundc_respond_err
00013164 C3                     <1>       retn
                                <1>
                                <1> snd_play_1:
00013165 E80C170000             <1>       call   vt8233_start_play
0001316A C3                     <1>       retn
                                <1>
                                <1> snd_play_2:
                                <1>       ; 28/05/2017
                                <1>       ;cmp   al, 2 ; AC'97
                                <1>       ;jne   short snd_play_3
                                <1>
0001316B E8D01C0000             <1>       call   ac97_start_play
00013170 C3                     <1>       retn
                                <1>
                                <1> ;snd_play_3:
                                <1> ;     ;call  hda_start_play
                                <1> ;     retn
                                <1>
                                <1> sound_pause:
                                <1>       ; FUNCTION = 5
                                <1>       ; Pause
                                <1>       ; 28/05/2017
```

```
                                <1>       ; 24/04/2017
                                <1>       ; 22/04/2017
00013171 E814030000             <1>       call   snd_dev_check
00013176 7275                   <1>       jc     short snd_nothing ; temporary.
00013178 E81A030000             <1>       call   snd_buf_check
0001317D 726E                   <1>       jc     short snd_nothing ; temporary.
0001317F A0[D98F0100]           <1>       mov    al, [audio_device]
00013184 3C03                   <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
00013186 7409                   <1>       je     short snd_pause_1
00013188 3C01                   <1>       cmp    al, 1 ; Sound Blaster 16
0001318A 750A                   <1>       jne    short snd_pause_2 ; 28/05/2017
0001318C E9941A0000             <1>       jmp    sb16_pause
                                <1> snd_pause_1:
00013191 E999170000             <1>       jmp    vt8233_pause
                                <1> snd_pause_2:
                                <1>       ; 28/05/2017
                                <1>       ;cmp   al, 2 ; AC'97
                                <1>       ;jne   short snd_nothing ; temporary.
00013196 E9331E0000             <1>       jmp    ac97_pause
                                <1>
                                <1> sound_continue:
                                <1>       ; FUNCTION = 6
                                <1>       ; Continue to play
                                <1>       ; 28/05/2017
                                <1>       ; 22/04/2017
0001319B E8EA020000             <1>       call   snd_dev_check
000131A0 724B                   <1>       jc     short snd_nothing ; temporary.
000131A2 E8F0020000             <1>       call   snd_buf_check
000131A7 7244                   <1>       jc     short snd_nothing ; temporary.
000131A9 A0[D98F0100]           <1>       mov    al, [audio_device]
000131AE 3C03                   <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
000131B0 7409                   <1>       je     short snd_cont_1
000131B2 3C01                   <1>       cmp    al, 1 ; Sound Blaster 16
000131B4 750A                   <1>       jne    short snd_cont_2 ; 28/05/2017
000131B6 E98D1A0000             <1>       jmp    sb16_continue
                                <1> snd_cont_1:
000131BB E920170000             <1>       jmp    vt8233_play
                                <1> snd_cont_2:
                                <1>       ; 28/05/2017
                                <1>       ;cmp   al, 2 ; AC'97
                                <1>       ;jne   short snd_nothing ; temporary.
000131C0 E9D11C0000             <1>       jmp    ac97_play
                                <1>
                                <1> sound_stop:
                                <1>       ; FUNCTION = 7
                                <1>       ; Stop playing
                                <1>       ; 28/05/2017
                                <1>       ; 24/05/2017
                                <1>       ; 21/04/2017, 22/04/2017, 24/04/2017
000131C5 E8C0020000             <1>       call   snd_dev_check
000131CA 7221                   <1>       jc     short snd_nothing ; temporary.
                                <1>       ;call  snd_buf_check
000131CC E8CF020000             <1>       call   snd_user_check ; 24/05/2017
000131D1 721A                   <1>       jc     short snd_nothing ; temporary.
                                <1>
000131D3 A0[D98F0100]           <1>       mov    al, [audio_device]
000131D8 3C03                   <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
000131DA 0F8456160000           <1>       je     vt8233_stop
                                <1>       ; 28/05/2017
                                <1>       ;ja    short snd_nothing
000131E0 3C01                   <1>       cmp    al, 1 ; Sound Blaster 16
000131E2 0F84831A0000           <1>       je     sb16_stop
                                <1>       ;cmp   al, 2
                                <1>       ;je    short ac97_stop
000131E8 E9B31D0000             <1>       jmp    ac97_stop ; temporary.
                                <1>       ;jmp   hda_stop
                                <1>
                                <1> snd_nothing:
                                <1>       ; 21/04/2017
000131ED C3                     <1>       retn
                                <1>
                                <1> soundc_reset:
                                <1>       ; FUNCTION = 8
                                <1>       ; Reset Audio Controller
                                <1>       ; 28/05/2017
                                <1>       ; 22/04/2017
000131EE E897020000             <1>       call   snd_dev_check
000131F3 72F8                   <1>       jc     snd_nothing ; temporary.
000131F5 E89D020000             <1>       call   snd_buf_check
000131FA 72F1                   <1>       jc     snd_nothing ; temporary.
                                <1>
000131FC A0[D98F0100]           <1>       mov    al, [audio_device]
00013201 3C03                   <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
00013203 0F8432170000           <1>       je     vt8233_reset
00013209 77E2                   <1>       ja     short snd_nothing ; temporary.
                                <1>       ;ja    hda_reset
0001320B 3C01                   <1>       cmp    al, 1 ; Sound Blaster 16
0001320D 0F850C1E0000           <1>       jne    ac97_reset
00013213 E8A51A0000             <1>       call   sb16_reset
00013218 0F822DFCFFFF           <1>       jc     soundc_respond_err
0001321E C3                     <1>       retn
                                <1>
                                <1> soundc_cancel:
                                <1>       ; FUNCTION = 9
                                <1>       ; Cancel audio callback service
                                <1>       ; 22/04/2017
0001321F A0[01900100]           <1>       mov    al, [audio_user]
00013224 3A05[B3030300]         <1>       cmp    al, [u.uno]
0001322A 75C1                   <1>       jne    short snd_nothing
                                <1>       ; RESET (audio) INTERRUPT CALLBACK SERVICE
0001322C 8A1D[DB8F0100]         <1>       mov    bl, [audio_intr] ; IRQ number
00013232 A0[B3030300]           <1>       mov    al, [u.uno]
00013237 28FF                   <1>       sub    bh, bh ; 0 ; unlink IRQ from user service
00013239 E8EC020000             <1>       call   set_irq_callback_service
0001323E 0F8250FDFFFF           <1>       jc     sndc_perm_error ; 'permission denied' error
```

```
5775 00013244 C3                <1>       retn
5776                            <1>
5777                            <1> sound_dalloc:
5778                            <1>       ; FUNCTION = 10
5779                            <1>       ; Deallocate (ring 3) audio buffer
5780                            <1>       ; 22/04/2017
5781 00013245 A0[01900100]      <1>       mov   al, [audio_user]
5782 0001324A 3A05[B3030300]    <1>       cmp   al, [u.uno]
5783 00013250 759B              <1>       jne   short snd_nothing
5784 00013252 8B1D[EC8F0100]    <1>       mov   ebx, [audio_buffer]
5785                            <1>       ;or   ebx, ebx
5786                            <1>       ;jz   short snd_nothing
5787 00013258 8B0D[F48F0100]    <1>       mov   ecx, [audio_buff_size]
5788 0001325E E87F30FFFF        <1>       call  deallocate_user_pages
5789 00013263 31C0              <1>       xor   eax, eax
5790 00013265 A3[EC8F0100]      <1>       mov   [audio_buffer], eax ; 0
5791 0001326A A2[01900100]      <1>       mov   [audio_user], al ; 0
5792 0001326F C3                <1>       retn
5793                            <1>
5794                            <1> sound_volume:
5795                            <1>       ; FUNCTION = 11
5796                            <1>       ; Set sound volume level
5797                            <1>       ; 28/05/2017
5798                            <1>       ; 20/05/2017
5799                            <1>       ; 22/04/2017, 24/04/2017
5800                            <1>       ; bl = component (0 = master/playback/lineout volume)
5801                            <1>       ; cl = left channel volume level (0 to 31)
5802                            <1>       ; ch = right channel volume level (0 to 31)
5803                            <1>
5804 00013270 80FB80            <1>       cmp   bl, 80h
5805 00013273 720E              <1>       jb    short snd_vol_1
5806 00013275 0F8772FFFFFF      <1>       ja    snd_nothing ; temporary.
5807                            <1>       ; Set volume level for next play (BL>= 80h)
5808 0001327B 66890D[0E900100]  <1>       mov   [audio_master_volume], cx
5809 00013282 C3                <1>       retn
5810                            <1> snd_vol_1:
5811                            <1>       ; set volume level immediate (BL< 80h)
5812 00013283 80FB00            <1>       cmp   bl, 0
5813 00013286 0F8761FFFFFF      <1>       ja    snd_nothing ; temporary.
5814                            <1>
5815 0001328C E8F9010000        <1>       call  snd_dev_check
5816 00013291 0F8256FFFFFF      <1>       jc    snd_nothing ; temporary.
5817 00013297 E8FB010000        <1>       call  snd_buf_check
5818 0001329C 0F824BFFFFFF      <1>       jc    snd_nothing ; temporary.
5819                            <1>
5820 000132A2 A0[D98F0100]      <1>       mov   al, [audio_device]
5821 000132A7 3C03              <1>       cmp   al, 3 ; VIA VT 8237R (vt8233)
5822 000132A9 0F84A5160000      <1>       je    vt8233_volume
5823                            <1>       ; 28/05/2017
5824 000132AF 0F8738FFFFFF      <1>       ja    snd_nothing ; temporary.
5825                            <1>       ;ja   hda_volume
5826                            <1>       ; Sound Blaster 16
5827 000132B5 3C01              <1>       cmp   al, 1 ; SB 16
5828 000132B7 0F8433190000      <1>       je    sb16_volume
5829 000132BD E9F01B0000        <1>       jmp   ac97_volume
5830                            <1>
5831                            <1> soundc_disable:
5832                            <1>       ; FUNCTION = 12
5833                            <1>       ; Disable audio device (and unlink DMA memory)
5834                            <1>       ; 28/05/2017
5835                            <1>       ; 24/05/2017
5836                            <1>       ; 22/04/2017
5837 000132C2 E8C3010000        <1>       call  snd_dev_check
5838 000132C7 0F8270FBFFFF      <1>       jc    soundc_dev_err ; temporary.
5839                            <1>       ;call snd_buf_check
5840                            <1>       ;jc   sndc_owner_error ; temporary.
5841                            <1>
5842 000132CD A0[D98F0100]      <1>       mov   al, [audio_device]
5843 000132D2 3C03              <1>       cmp   al, 3 ; VIA VT 8237R (vt8233)
5844 000132D4 7418              <1>       je    short snd_disable_1
5845 000132D6 0F8711FFFFFF      <1>       ja    snd_nothing ; temporary.
5846 000132DC 3C01              <1>       cmp   al, 1 ; Sound Blaster 16
5847 000132DE 7507              <1>       jne   short snd_disable_0
5848 000132E0 E886190000        <1>       call  sb16_stop
5849 000132E5 EB0C              <1>       jmp   short snd_disable_2
5850                            <1> snd_disable_0:
5851 000132E7 E8B41C0000        <1>       call  ac97_stop
5852 000132EC EB05              <1>       jmp   short snd_disable_2
5853                            <1> snd_disable_1:
5854 000132EE E843150000        <1>       call  vt8233_stop
5855                            <1> snd_disable_2:
5856 000132F3 A0[DB8F0100]      <1>       mov   al, [audio_intr]
5857 000132F8 29DB              <1>       sub   ebx, ebx ; 0 = reset
5858 000132FA E851FAFFFF        <1>       call  set_dev_IRQ_service
5859                            <1>
5860                            <1>       ;mov  al, [audio_intr]
5861 000132FF 28E4              <1>       sub   ah, ah ; 0 = reset
5862 00013301 E801F8FFFF        <1>       call  set_hardware_int_vector
5863                            <1>
5864 00013306 31C0              <1>       xor   eax, eax
5865 00013308 A2[D98F0100]      <1>       mov   byte [audio_device], al
5866 0001330D A2[DB8F0100]      <1>       mov   byte [audio_intr], al
5867 00013312 8705[F88F0100]    <1>       xchg  eax, [audio_dma_buff]
5868                            <1>       ; 24/05/2017
5869                            <1>       ;or   eax, eax
5870                            <1>       ;jz   short snd_disable_3
5871                            <1>       ;cmp  eax, sb16_dma_buffer ; default DMA buffer
5872                            <1>       ;je   short snd_disable_3
5873 00013318 803D[D88F0100]00  <1>       cmp   byte [audio_pci], 0 ; AC97 audio controller ?
5874 0001331F 7612              <1>       jna   short snd_disable_3
5875 00013321 C605[D88F0100]00  <1>       mov   byte [audio_pci], 0
5876                            <1>       ;sub  ecx, ecx
5877                            <1>       ;xchg ecx, [audio_dmabuff_size]
5878 00013328 8B0D[FC8F0100]    <1>       mov   ecx, [audio_dmabuff_size]
5879 0001332E E8712EFFFF        <1>       call  deallocate_memory_block
```

```
5880                                    <1> snd_disable_3:
5881 00013333 C3                        <1>         retn
5882                                    <1>
5883                                    <1> sound_dma_map:
5884                                    <1>         ; FUNCTION = 13
5885                                    <1>         ; Map audio dma buff addr to user's buffer addr
5886                                    <1>         ; 12/05/2017
5887 00013334 21C9                      <1>         and    ecx, ecx
5888 00013336 0F8408FBFFFF              <1>         jz     sound_buff_error
5889 0001333C 803D[D98F0100]01          <1>         cmp    byte [audio_device], 1
5890 00013343 7229                      <1>         jb     short snd_dma_map_1
5891                                    <1> snd_dma_map_0:
5892 00013345 A1[F88F0100]              <1>         mov    eax, [audio_dma_buff]
5893 0001334A 21C0                      <1>         and    eax, eax
5894 0001334C 7420                      <1>         jz     short snd_dma_map_1
5895                                    <1>         ;
5896 0001334E 8A1D[01900100]            <1>         mov    bl, [audio_user]
5897 00013354 08DB                      <1>         or     bl, bl
5898 00013356 7416                      <1>         jz     short snd_dma_map_1
5899 00013358 3A1D[B3030300]            <1>         cmp    bl, [u.uno]
5900 0001335E 0F852BFCFFFF              <1>         jne    sndc_owner_error
5901                                    <1>         ;
5902 00013364 8B1D[FC8F0100]            <1>         mov    ebx, [audio_dmabuff_size]
5903 0001336A 21DB                      <1>         and    ebx, ebx
5904 0001336C 750A                      <1>         jnz    short snd_dma_map_2
5905                                    <1> snd_dma_map_1:
5906 0001336E B8[00000200]              <1>         mov    eax, sb16_dma_buffer
5907 00013373 BB00000100                <1>         mov    ebx, 65536
5908                                    <1> snd_dma_map_2:
5909 00013378 81C1FF0F0000              <1>         add    ecx, PAGE_SIZE-1 ; 4095
5910 0001337E 6681E100F0                <1>         and    cx, ~PAGE_OFF ; not 4095
5911 00013383 39D9                      <1>         cmp    ecx, ebx
5912 00013385 0F87B9FAFFFF              <1>         ja     sound_buff_error
5913 0001338B 50                        <1>         push   eax
5914 0001338C 89D3                      <1>         mov    ebx, edx
5915 0001338E C1E90C                    <1>         shr    ecx, 12 ; byte count to page count
5916                                    <1>         ; eax = physical address of (audio) dma buffer
5917                                    <1>         ; ebx = virtual address of (audio) dma buffer (user's pgdir)
5918                                    <1>         ; ecx = page count (>0)
5919 00013391 E87E2EFFFF                <1>         call   direct_memory_access
5920 00013396 58                        <1>         pop    eax
5921 00013397 0F82A7FAFFFF              <1>         jc     sound_buff_error
5922 0001339D A3[64030300]              <1>         mov    [u.r0], eax
5923 000133A2 C3                        <1>         retn
5924                                    <1>
5925                                    <1> soundc_info:
5926                                    <1>         ; FUNCTION = 14
5927                                    <1>         ; Get Audio Controller Info
5928                                    <1>         ; 10/06/2017
5929                                    <1>         ; 05/06/2017
5930 000133A3 20DB                      <1>         and    bl, bl ; 0
5931 000133A5 740A                      <1>         jz     short sndc_info_0
5932                                    <1>         ; invalid parameter !
5933 000133A7 B817000000                <1>         mov    eax, ERR_INV_PARAMETER ; 23
5934                                    <1> ;sndc_inf_error:
5935                                    <1> ;       mov    [u.r0], eax
5936                                    <1> ;       mov    [u.error], eax
5937                                    <1> ;       jmp    error
5938 000133AC E99FFAFFFF                <1>         jmp    sysaudio_err
5939                                    <1>
5940                                    <1> sndc_info_0:
5941 000133B1 E8D4000000                <1>         call   snd_dev_check
5942 000133B6 0F8281FAFFFF              <1>         jc     soundc_dev_err
5943                                    <1>
5944 000133BC 8B1D[E48F0100]            <1>         mov    ebx, [audio_vendor]
5945 000133C2 8B0D[E08F0100]            <1>         mov    ecx, [audio_dev_id]
5946                                    <1>         ;mov   al,  [audio_device]
5947 000133C8 3C02                      <1>         cmp    al, 2 ; AC'97 (ICH)
5948 000133CA 7513                      <1>         jne    short sndc_info_1
5949                                    <1>         ; Intel AC97 (ICH) Audio Controller (=2)
5950 000133CC 668B15[12900100]          <1>         mov    dx, [NABMBAR]
5951 000133D3 C1E210                    <1>         shl    edx, 16
5952 000133D6 668B15[10900100]          <1>         mov    dx, [NAMBAR]
5953 000133DD EB07                      <1>         jmp    short sndc_info_2
5954                                    <1> sndc_info_1:
5955                                    <1>         ; 05/06/2017
5956                                    <1>         ; Note: Intel HDA code (here) is not ready yet!
5957                                    <1>         ; !!! SB16 or VT8233 (VT8237R) !!!
5958 000133DF 0FB715[DE8F0100]          <1>         movzx  edx, word [audio_io_base]
5959                                    <1> sndc_info_2:
5960 000133E6 88C4                      <1>         mov    ah, al ;  [audio_device]
5961 000133E8 A0[DB8F0100]              <1>         mov    al, [audio_intr]
5962                                    <1>
5963                                    <1>         ; EAX = IRQ Number in AL
5964                                    <1>         ;       Audio Device Number in AH
5965                                    <1>         ; EBX = DEV/VENDOR ID
5966                                    <1>         ;       (DDDDDDDDDDDDDDDDVVVVVVVVVVVVVVVV)
5967                                    <1>         ; ECX = BUS/DEV/FN
5968                                    <1>         ;       (00000000BBBBBBBBDDDDDFFF00000000)
5969                                    <1>         ; EDX = NABMBAR/NAMBAR (for AC97)
5970                                    <1>         ;       (Low word, DX = NAMBAR address)
5971                                    <1>         ; EDX = Base IO Addr (DX) for SB16 & VT8233
5972                                    <1>
5973                                    <1>         ; 10/06/2017
5974 000133ED A3[64030300]              <1>         mov    [u.r0], eax
5975 000133F2 8B2D[60030300]            <1>         mov    ebp, [u.usp]
5976 000133F8 895D10                    <1>         mov    [ebp+16], ebx  ; ebx
5977 000133FB 895514                    <1>         mov    [ebp+20], edx  ; edx
5978 000133FE 894D18                    <1>         mov    [ebp+24], ecx  ; ecx
5979                                    <1>
5980 00013401 C3                        <1>         retn
5981                                    <1>
5982                                    <1> sound_data:
5983                                    <1>         ; FUNCTION = 15
5984                                    <1>         ; Get Current Sound data for graphics
```

```
                                <1>       ; 22/06/2017
                                <1>       ;
00013402 E883000000             <1>       call   snd_dev_check
00013407 0F8230FAFFFF           <1>       jc     soundc_dev_err ; Device not ready !
                                <1>
0001340D 80FB00                 <1>       cmp    bl, 0
00013410 760A                   <1>       jna    short sound_data_0
                                <1>
                                <1>       ; Only PCM OUT buffer data is valid for now!
00013412 B817000000             <1>       mov    eax, ERR_INV_PARAMETER  ; 23
00013417 E934FAFFFF             <1>       jmp    sysaudio_err
                                <1>
                                <1> sound_data_0:
0001341C A1[F88F0100]           <1>       mov    eax, [audio_dma_buff]
00013421 09C0                   <1>       or     eax, eax
00013423 0F841BFAFFFF           <1>       jz     sound_buff_error
                                <1>
00013429 803D[D98F0100]04       <1>       cmp    byte [audio_device], 4 ; Intel HDA
00013430 744F                   <1>       je     short sound_data_4 ; temporary ! (22/06/2017)
                                <1>
00013432 21C9                   <1>       and    ecx, ecx
                                <1>       ;jnz   short sound_data_1 ; sample tranfer
                                <1>
                                <1>       ; Return only DMA Buffer pointer/offset...
                                <1>       ; (If DMA Buffer has been mapped to user's
                                <1>       ;  memory space; program can get graphics
                                <1>       ;  data by using only this pointer value.)
                                <1>
                                <1>       ;call  get_dma_buffer_offset
                                <1>       ;; eax = DMA buffer offset
                                <1>       ;;     (!not half buffer offset!)
                                <1>       ;mov   [u.r0], eax
                                <1>       ;retn
                                <1>
00013434 0F845D1D0000           <1>       jz     get_dma_buffer_offset
                                <1>
                                <1> sound_data_1:
                                <1>       ;mov   eax, [audio_dmabuff_size]
                                <1>       ;shr   eax, 1 ; half buffer size
                                <1>       ;cmp   ecx, eax
                                <1>       ;ja    short sound_buff_error
                                <1>
0001343A 3B0D[FC8F0100]         <1>       cmp    ecx, [audio_dmabuff_size]
00013440 0F87FEF9FFFF           <1>       ja     sound_buff_error
                                <1>
00013446 89D0                   <1>       mov    eax, edx
00013448 25FF0F0000             <1>       and    eax, PAGE_OFF ; 4095 (0FFFh)
0001344D 81F900100000           <1>       cmp    ecx, 4096
00013453 7605                   <1>       jna    short sound_data_2
00013455 B900100000             <1>       mov    ecx, 4096 ; max. 1 page
                                <1> sound_data_2:
0001345A 01C8                   <1>       add    eax, ecx
0001345C 3D00100000             <1>       cmp    eax, 4096
00013461 7606                   <1>       jna    short sound_data_3
00013463 6625FF0F               <1>       and    ax, PAGE_OFF ; 4095 (0FFFh)
00013467 29C1                   <1>       sub    ecx, eax
                                <1>       ; here, ECX has been adjusted to fit
                                <1>       ;      in page border..  (<= 4096, >0)
                                <1> sound_data_3:
00013469 51                     <1>       push   ecx
0001346A 52                     <1>       push   edx
0001346B 89D3                   <1>       mov    ebx, edx
0001346D E8F029FFFF             <1>       call   get_physical_addr
00013472 5A                     <1>       pop    edx
00013473 59                     <1>       pop    ecx
00013474 0F82CAF9FFFF           <1>       jc     sound_buff_error
                                <1>
                                <1>       ; eax = physical address of user's buffer
0001347A 89C3                   <1>       mov    ebx, eax
                                <1>       ; ecx = byte (transfer) count
                                <1>       ;call  get_current_sound_data
                                <1>       ;retn
0001347C E9731C0000             <1>       jmp    get_current_sound_data
                                <1>
                                <1> sound_data_4:
                                <1>       ; Intel HDA code is not ready yet !
                                <1>       ; 22/06/2017
00013481 31C0                   <1>       xor    eax, eax
00013483 48                     <1>       dec    eax
00013484 A3[64030300]           <1>       mov    [u.r0], eax ; 0FFFFFFFFh
00013489 C3                     <1>       retn
                                <1>
                                <1> snd_dev_check:
                                <1>       ; 10/06/2017
                                <1>       ; 05/06/2017
                                <1>       ; 24/05/2017
                                <1>       ; 22/04/2017
                                <1>       ; 21/04/2017
                                <1>       ; ... device check at first
0001348A A0[D98F0100]           <1>       mov    al, [audio_device]
0001348F 3C01                   <1>       cmp    al, 1 ; SB 16
00013491 7203                   <1>       jb     short snd_dev_chk_retn ; error !
                                <1>       ;cmp   al, 4 ; Intel HDA
                                <1>       ;ja    short snd_dbchk_stc ; invalid !
                                <1>       ; 10/06/2017
00013493 3C05                   <1>       cmp    al, 5
00013495 F5                     <1>       cmc
                                <1> snd_dev_chk_retn:
00013496 C3                     <1>       retn
                                <1>
                                <1> snd_buf_check:
                                <1>       ; 10/06/2017
                                <1>       ; 22/04/2017
                                <1>       ; 21/04/2017
                                <1>       ; ... buffer & (buffer) owner check at second
```

```
6090 00013497 833D[EC8F0100]00    <1>       cmp    dword [audio_buffer], 0
6091 0001349E 760D                <1>       jna    short snd_dbchk_stc
6092                              <1> snd_user_check:
6093 000134A0 A0[B3030300]        <1>       mov    al, [u.uno]
6094 000134A5 3A05[01900100]      <1>       cmp    al, [audio_user]
6095                              <1>       ;jne   short snd_dbchk_stc
6096                              <1>       ;retn
6097 000134AB 74E9                <1>       je     short snd_dev_chk_retn
6098                              <1>
6099                              <1> snd_dbchk_stc:
6100 000134AD F9                  <1>       stc
6101 000134AE C3                  <1>       retn
6102                              <1>
6103                              <1> sound_update:
6104                              <1>       ; FUNCTION = 16
6105                              <1>       ; bl =
6106                              <1>       ;    0 = automatic (sequental) update (with flag switch!)
6107                              <1>       ;    1 = update dma half buffer 1 (without flag switch!)
6108                              <1>       ;    2 = update dma half buffer 2 (without flag switch!)
6109                              <1>       ;  FFh = get current flag value
6110                              <1>       ;      0 = dma half buffer 1 (will be played next)
6111                              <1>       ;      1 = dma half buffer 2 (will be played next)
6112                              <1>
6113                              <1>       ; 10/10/2017
6114                              <1>
6115                              <1>       ; ... device check at first
6116 000134AF A0[D98F0100]        <1>       mov    al, [audio_device]
6117 000134B4 08C0                <1>       or     al, al ; 0 ; pc speaker or invalid
6118 000134B6 0F8481F9FFFF        <1>       jz     soundc_dev_err
6119                              <1>
6120                              <1>       ; ... buffer & (buffer) owner check at second
6121 000134BC 833D[EC8F0100]00    <1>       cmp    dword [audio_buffer], 0
6122 000134C3 0F867BF9FFFF        <1>       jna    sound_buff_error
6123 000134C9 A0[B3030300]        <1>       mov    al, [u.uno]
6124 000134CE 3A05[01900100]      <1>       cmp    al, [audio_user]
6125 000134D4 0F85B5FAFFFF        <1>       jne    sndc_owner_error
6126                              <1>
6127                              <1>       ; Transfer ring 3 (user's) audio buffer content to dma buffer
6128 000134DA 8B3D[F88F0100]      <1>       mov    edi, [audio_dma_buff] ; dma buffer (ring 0)
6129 000134E0 09FF                <1>       or     edi, edi
6130 000134E2 0F845CF9FFFF        <1>       jz     sound_buff_error
6131 000134E8 8B35[F08F0100]      <1>       mov    esi, [audio_p_buffer] ; physical address (ring 3)
6132 000134EE 8B0D[F48F0100]      <1>       mov    ecx, [audio_buff_size]
6133                              <1>
6134                              <1>       ;movzx eax, byte [audio_flag]
6135 000134F4 A0[00900100]        <1>       mov    al, [audio_flag]
6136                              <1>
6137 000134F9 FEC3                <1>       inc    bl
6138 000134FB 7427                <1>       jz     short snd_update_3 ; bl = 0FFh
6139 000134FD FECB                <1>       dec    bl
6140 000134FF 7411                <1>       jz     short snd_update_0 ; bl = 0
6141                              <1>
6142 00013501 80FB02              <1>       cmp    bl, 2
6143 00013504 7417                <1>       je     short snd_update_1 ; dma half buffer 2
6144 00013506 7217                <1>       jb     short snd_update_2 ; dma half buffer 1
6145                              <1>
6146                              <1>       ; invalid parameter !
6147 00013508 B817000000          <1>       mov    eax, ERR_INV_PARAMETER ; 23
6148                              <1> ;     mov    [u.r0], eax
6149                              <1> ;     mov    [u.error], eax
6150                              <1> ;     jmp    error
6151 0001350D E93EF9FFFF          <1>       jmp    sysaudio_err
6152                              <1>
6153                              <1> snd_update_0:
6154 00013512 8035[00900100]01    <1>       xor    byte [audio_flag], 1 ; update flag !!!
6155 00013519 3C01                <1>       cmp    al, 1
6156 0001351B 7202                <1>       jb     short snd_update_2 ; dma half buffer 1
6157                              <1> snd_update_1:
6158                              <1>       ; dma half buffer 2
6159 0001351D 01CF                <1>       add    edi, ecx
6160                              <1> snd_update_2:
6161                              <1>       ;rep   movsb
6162 0001351F C1E902              <1>       shr    ecx, 2
6163 00013522 F3A5                <1>       rep    movsd
6164                              <1> snd_update_3:
6165 00013524 A3[64030300]        <1>       mov    [u.r0], eax
6166                              <1>
6167 00013529 C3                  <1>       retn
6168                              <1>
6169                              <1> set_irq_callback_service:
6170                              <1>       ; 03/08/2020
6171                              <1>       ; 10/06/2017
6172                              <1>       ; 12/05/2017
6173                              <1>       ; 24/04/2017
6174                              <1>       ; 22/04/2017
6175                              <1>       ; caller: 'syscalbac' or 'sysaudio' or ...
6176                              <1>       ; 13/04/2017, 14/04/2017, 17/04/2017
6177                              <1>       ; 24/02/2017, 26/02/2017, 28/02/2017
6178                              <1>       ; 21/02/2017 - TRDOS 386 (TRDOS v2.0)
6179                              <1>       ;
6180                              <1>       ; Link or unlink IRQ callback service to/from user (ring 3)
6181                              <1>       ;
6182                              <1>       ; INPUT ->
6183                              <1>       ;     If AL = 0, the caller is 'syscalbac';
6184                              <1>       ;        otherwise, the caller is 'sysaudio' or ...
6185                              <1>       ;        (AL = user number)
6186                              <1>       ;
6187                              <1>       ;     BL = IRQ number (Hardware interrupt request number)
6188                              <1>       ;          (0 t0 15 but IRQ 0,1,2,6,8,14,15 are prohibited)
6189                              <1>       ;          IRQ numbers 3,4,5,7,9,10,11,12,13 are valid
6190                              <1>       ;          (numbers >15 are invalid)
6191                              <1>       ;
6192                              <1>       ;     BH = 0 = Unlink IRQ (in BL) from user (ring 3) service
6193                              <1>       ;          1 = Link IRQ by using Signal Response Byte method
6194                              <1>       ;          2 = Link IRQ by using Callback service method
```

```
6195                                 <1>       ;          3 = Link IRQ by using Auto Increment S.R.B. method
6196                                 <1>       ;         >3 = invalid
6197                                 <1>       ;             (syscallback version will return to user)
6198                                 <1>       ;
6199                                 <1>       ;     CL = Signal Return/Response Byte value
6200                                 <1>       ;
6201                                 <1>       ;     If BH = 3, kernel will put a counter value ; 03/08/2020
6202                                 <1>       ;               (into the S.R.B. addr)
6203                                 <1>       ;              between 0 to 255. (start value = CL+1)
6204                                 <1>       ;
6205                                 <1>       ;     NOTE: counter value, for example: even and odd numbers
6206                                 <1>       ;           may be used for -audio- DMA buffer switch
6207                                 <1>       ;           within double buffer method, etc.
6208                                 <1>       ;
6209                                 <1>       ;     EDX = Signal return (Response) byte address
6210                                 <1>       ;                         - or -
6211                                 <1>       ;           Interrupt/Callback service/routine address
6212                                 <1>       ;
6213                                 <1>       ;           (virtual address in user's memory space)
6214                                 <1>       ;
6215                                 <1>       ; OUTPUT ->
6216                                 <1>       ;     CF = 0 & EAX = 0 -> Successful setting
6217                                 <1>       ;     CF = 1 & EAX > 0 -> IRQ is prohibited or locked
6218                                 <1>       ;                   by another process
6219                                 <1>       ;             eax = ERR_PERM_DENIED -> prohibited or locked
6220                                 <1>       ;             eax = ERR_INV_PARAMETER ->
6221                                 <1>       ;                   invalid parameter/option or bad address
6222                                 <1>       ;
6223                                 <1>       ; TRDOS 386 - IRQ CALLBACK structures (parameters):
6224                                 <1>       ;
6225                                 <1>       ;        [u.irqlock] = 1 word, IRQ flags (0-15) that indicates
6226                                 <1>       ;                   which IRQs are locked by (that) user.
6227                                 <1>       ;                    Lock and unlock (by user) will change
6228                                 <1>       ;                   these flags or 'terminate process' (sysexit)
6229                                 <1>       ;                   will clear these flags and unlock those IRQs.
6230                                 <1>       ;
6231                                 <1>       ;                   Bit 0 is for IRQ 0 and Bit 15 is for IRQ 15
6232                                 <1>       ;
6233                                 <1>       ;        IRQ(x).owner      : 1 byte, user, [u.uno], 0 = free (unlocked)
6234                                 <1>       ;
6235                                 <1>       ;        IRQ(x).method : 1 byte for callback method & status
6236                                 <1>       ;                     0 = Signal Response Byte method
6237                                 <1>       ;                     1 = Callback service method
6238                                 <1>       ;                     >1 = invalid for current 'syscalback'.
6239                                 <1>       ;                   or(+) 80h = IRQ is in use by system (ring 0)
6240                                 <1>       ;                               function (audio etc.) or
6241                                 <1>       ;                             a device driver.
6242                                 <1>       ;                   (system function will ignore the lock/owner)
6243                                 <1>       ;
6244                                 <1>       ;        IRQ(x).srb: 1 byte, Signal Return/Response byte value
6245                                 <1>       ;                     (a fixed value by user or a counter value
6246                                 <1>       ;                    from 0 to 255, which is increased by every
6247                                 <1>       ;                    interrupt just before putting it into
6248                                 <1>       ;                    the Signal Response byte address
6249                                 <1>       ;                    (This is not used in callback serv method)
6250                                 <1>       ;
6251                                 <1>       ;        IRQ(x).addr       : 1 dword
6252                                 <1>       ;                     Signal Response Byte address (physical)
6253                                 <1>       ;                                -or-
6254                                 <1>       ;                     Callback service address (virtual)
6255                                 <1>       ;
6256                                 <1>       ;        IRQ(x).dev: 1 byte
6257                                 <1>       ;                     0 = Default device or kernel function
6258                                 <1>       ;                                -or-
6259                                 <1>       ;                     1-255 = Assigned device driver number
6260                                 <1>       ;
6261                                 <1>       ;        (x) = 3,4,5,7,9,10,11,12,13
6262                                 <1>       ;
6263                                 <1>
6264 0001352A 80FB0F                <1>       cmp    bl, 15
6265 0001352D 7729                  <1>       ja     short scbs_2
6266                                 <1>
6267 0001352F 80FF03                <1>       cmp    bh, 3
6268 00013532 7724                  <1>       ja     short scbs_2  ; invalid parameter
6269                                 <1>
6270 00013534 0FB6FB                <1>       movzx  edi, bl ; save IRQ number
6271                                 <1>
6272                                 <1>              ;  IRQ 0,1,2,6,8,14,15 are prohibited
6273                                 <1>       ;IRQenum: ; 'trdosk9.s'
6274                                 <1>       ;      db  0,0,0,1,2,3,0,4,0,5,6,7,8,9,0,0
6275                                 <1>
6276 00013537 0FB6B7[70400100]      <1>       movzx  esi, byte [edi+IRQenum] ; IRQ availability
6277                                 <1>                                      ; enumeration/index
6278                                 <1>       ;dec   esi
6279 0001353E 664E                  <1>       dec    si
6280 00013540 780F                  <1>       js     short scbs_1 ;  0 -> 0FFFFh
6281                                 <1>
6282                                 <1>       ; ESI = IRQ callback parameters index number (0 to 8)
6283                                 <1>
6284 00013542 08FF                  <1>       or     bh, bh
6285 00013544 7419                  <1>       jz     short scbs_4 ; unlink the IRQ (in BL)
6286                                 <1>
6287 00013546 FECF                  <1>       dec    bh
6288                                 <1>       ; bh = method (0 = signal response byte, 1 = callback)
6289                                 <1>       ;            (2 = auto increment of signal response byte)
6290                                 <1>
6291 00013548 80BE[888F0100]00      <1>       cmp    byte [esi+IRQ.owner], 0 ; locked ?
6292 0001354F 7637                  <1>       jna    short scbs_6 ; no... OK...
6293                                 <1>
6294                                 <1> scbs_1:
6295                                 <1>       ; permission denied (prohibited IRQ)
6296 00013551 B80B000000            <1>       mov    eax, ERR_PERM_DENIED
6297 00013556 F9                    <1>       stc
6298 00013557 C3                    <1>       retn
6299                                 <1> scbs_2:
```

```
6300 00013558 F9                <1>       stc
6301                            <1> scbs_3:
6302 00013559 B817000000        <1>       mov    eax, ERR_INV_PARAMETER
6303 0001355E C3                <1>       retn
6304                            <1>
6305                            <1> scbs_4: ; unlink the requested IRQ (if it belongs to current user)
6306                            <1>       ; 10/06/2017
6307                            <1>       ; 22/04/2017
6308                            <1>       ; 14/04/2017
6309                            <1>       ; If AL = 0 -> The caller is 'syscalbac'
6310 0001355F 8AA6[888F0100]    <1>       mov    ah, [esi+IRQ.owner]
6311 00013565 3A25[B3030300]    <1>       cmp    ah, [u.uno]
6312 0001356B 75E4              <1>       jne    short scbs_1
6313                            <1>
6314 0001356D FE0D[D6030300]    <1>       dec    byte [u.irqc] ; decrease IRQ count (in use)
6315                            <1>
6316                            <1>       ;sub   ah, ah
6317                            <1>       ;mov   [esi+IRQ.owner], ah ; 0 ; free !!!
6318                            <1>       ;and   byte [esi+IRQ.method], 80h
6319                            <1>       ;mov   [esi+IRQ.srb], ah ; 0
6320                            <1>       ;mov   [esi+IRQ.dev], ah ; 0
6321                            <1>       ;mov   dword [esi+IRQ.addr], 0
6322                            <1>       ;mov   dword [u.r0], 0
6323                            <1>
6324                            <1>       ;mov   byte [esi+IRQ.owner], 0
6325                            <1>
6326                            <1>       ; 22/04/2017
6327 00013573 29C0              <1>       sub    eax, eax
6328 00013575 8886[888F0100]    <1>       mov    [esi+IRQ.owner], al ; 0
6329                            <1>       ; 10/06/2017
6330 0001357B 8686[9A8F0100]    <1>       xchg   al, [esi+IRQ.method]
6331 00013581 2480              <1>       and    al, 80h
6332 00013583 745E              <1>       jz     short scbs_12
6333                            <1>       ; Audio device must be disabled -later- ! ([IRQ.medhod] = 80h)
6334                            <1>
6335                            <1> ;     cmp    byte [esi+IRQ.method], 80h ; device drv or kernel extension ?
6336                            <1> ;     jb     short scbs_12 ; bh = 0 reset to default IRQ handler
6337                            <1> ;
6338                            <1> ;     and    al, al
6339                            <1> ;     jz     short  scbs_5  ; the caller is 'syscalbac'
6340                            <1> ;     ; The caller is 'sysaudio' or ...
6341 00013585 30C0              <1>       xor    al, al
6342                            <1> ;     mov    [esi+IRQ.method], al ; 0 ; reset kernel extension flag
6343                            <1> ;scbs_5:
6344                            <1> ;     sub    ah, ah
6345                            <1>       ;mov   [u.r0], eax ; 0
6346 00013587 C3                <1>       retn
6347                            <1>
6348                            <1> scbs_6:
6349                            <1>       ; 14/04/2017
6350 00013588 20C0              <1>       and    al, al
6351 0001358A 7405              <1>       jz     short scbs_7  ; the caller is 'syscalbac'
6352                            <1>       ; AL = user number ([u.uno] or [audio.user] or ...)
6353                            <1>       ; The caller is 'sysaudio' or ...
6354                            <1>       ;
6355                            <1>       ; bh = method (0 = signal response byte, 1 = callback)
6356                            <1>       ;            (2 = auto increment of signal response byte)
6357                            <1>
6358 0001358C 80CF80            <1>       or     bh, 80h             ; Kernel extension flag !
6359 0001358F EB0A              <1>       jmp    short scbs_8
6360                            <1> scbs_7:
6361 00013591 8A86[9A8F0100]    <1>       mov    al, [esi+IRQ.method] ; >= 80h = kernel is using this IRQ
6362 00013597 2480              <1>       and    al, 80h ; use only bit 7 (kernel function flag)
6363 00013599 08C7              <1>       or     bh, al        ; method
6364                            <1>                            ; 0 = signal response byte, 1 = callback
6365                            <1>                            ; 2 = auto increment of s.r.b.
6366                            <1> scbs_8:
6367 0001359B A0[B3030300]      <1>       mov    al, [u.uno] ; user (process) number (1 to 16)
6368 000135A0 8886[888F0100]    <1>       mov    [esi+IRQ.owner], al ; lock the IRQ for user
6369 000135A6 88BE[9A8F0100]    <1>       mov    [esi+IRQ.method], bh
6370                            <1>
6371                            <1> ;     test   bh, 1
6372                            <1> ;     jnz    short scbs_9  ; Callback method, CX will not be used
6373                            <1> ;
6374                            <1> ;     test   bh, 2         ; use auto increment (counter) method
6375                            <1> ;     jz     short scbs_10 ; (count can be used for buffer switch)
6376                            <1> ;scbs_9:
6377                            <1> ;     xor    ecx, ecx ; 0
6378                            <1> scbs_10:
6379                            <1>       ;mov   [esi+IRQ.method], bh
6380 000135AC 888E[A38F0100]    <1>       mov    [esi+IRQ.srb], cl
6381 000135B2 C686[918F0100]00  <1>       mov    byte [esi+IRQ.dev], 0 ; device number is always 0
6382                            <1>                            ; for this system call
6383                            <1>       ;test  bh, 1
6384 000135B9 80E701            <1>       and    bh, 1 ; 17/04/2017
6385 000135BC 7513              <1>       jnz    short scbs_11 ; callback method, use virtual address
6386                            <1>
6387 000135BE 53                <1>       push   ebx ; IRQ number (in BL)
6388 000135BF 89D3              <1>       mov    ebx, edx
6389                            <1>       ; ebx = virtual address
6390                            <1>       ; [u.pgdir] = page directory's physical address
6391 000135C1 FE05[468F0100]    <1>       inc     byte [no_page_swap] ; 1
6392                            <1>                    ; Do not add this page to swap queue
6393                            <1>                    ; and remove it from swap queue if it is
6394                            <1>                    ; on the queue.
6395 000135C7 E89628FFFF        <1>       call   get_physical_addr
6396 000135CC 5B                <1>       pop    ebx
6397 000135CD 728A              <1>       jc     scbs_3 ; invalid address !
6398                            <1>       ; eax = physical address of the virtual address in user's space
6399 000135CF 89C2              <1>       mov    edx, eax
6400                            <1> scbs_11:
6401 000135D1 66C1E602          <1>       shl    si, 2        ; byte (index) to dword (offset)
6402 000135D5 8996[AC8F0100]    <1>       mov    [esi+IRQ.addr], edx
6403                            <1>
6404 000135DB FE05[D6030300]    <1>       inc    byte [u.irqc]; increase IRQ (in use) count
```

```
6405                                    <1>
6406 000135E1 FEC7                      <1>       inc     bh ; 17/04/2017
6407                                    <1>       ; bh > 0 -> set to requested IRQ handler (IRQ_u_list)
6408                                    <1> scbs_12:
6409 000135E3 88D8                      <1>       mov     al, bl ; IRQ number
6410 000135E5 88FC                      <1>       mov     ah, bh ; 0 = reset, >0 = set
6411 000135E7 E81BF5FFFF                <1>       call    set_hardware_int_vector
6412                                    <1>
6413 000135EC 31C0                      <1>       xor     eax, eax
6414                                    <1>       ;mov    [u.r0], eax ; 0
6415                                    <1>
6416 000135EE C3                        <1>       retn    ; return with success (cf=0, eax=0)
6417                                    <1>
6418                                    <1>
6419                                    <1> sysdma: ; DMA FUNCTIONS
6420                                    <1>       ; 02/09/2017
6421                                    <1>       ; 28/08/2017
6422                                    <1>       ; 20/08/2017 - TRDOS 386 (TRDOS v2.0)
6423                                    <1>       ;
6424                                    <1>       ; Inputs:
6425                                    <1>       ;      BH = 0 -> Allocate DMA buffer
6426                                    <1>       ;         BL = 0 -> Use the system's default DMA
6427                                    <1>       ;                   (SB16) Buffer
6428                                    <1>       ;              Buffer Size (max.) = 65536 bytes
6429                                    <1>       ;         BL > 0 -> Allocate (a new) DMA buffer
6430                                    <1>       ;         ECX = DMA Buffer Size in bytes (<=128KB)
6431                                    <1>       ;         EDX = Virtual Address of DMA buffer
6432                                    <1>       ;
6433                                    <1>       ;      BH = 1 -> Initialize (Start) DMA service
6434                                    <1>       ;           BL, bit 0 to 3 = Channel Number (0 to 7)
6435                                    <1>       ;           BL, bit 7 = Auto Initialized Mode
6436                                    <1>       ;                    (If bit 7 is set)
6437                                    <1>       ;              bit 6 = Record (read) mode (0= playback)
6438                                    <1>       ;           ECX = byte count (0 = use dma buffer size)
6439                                    <1>       ;           EDX = physical buffer address
6440                                    <1>       ;                (0 = use dma buffer -start- address)
6441                                    <1>       ;
6442                                    <1>       ;      BH = 2 -> Get Current DMA Buffer Offset
6443                                    <1>       ;           BL = DMA channel number
6444                                    <1>       ;
6445                                    <1>       ;      BH = 3 -> Get Current DMA count down value
6446                                    <1>       ;           BL = DMA channel number (0 tO 7)
6447                                    <1>       ;
6448                                    <1>       ;      BH = 4 -> Get Current DMA channel (in progress)
6449                                    <1>       ;
6450                                    <1>       ;      BH = 5 -> Get System's Default DMA Buffer Address
6451                                    <1>       ;
6452                                    <1>       ;      BH = 6 -> Get Current DMA Buffer Address
6453                                    <1>       ;
6454                                    <1>       ;      BH = 7 -> Stop DMA service
6455                                    <1>       ;
6456                                    <1>       ;
6457                                    <1>       ; Outputs:
6458                                    <1>       ;
6459                                    <1>       ;      For BH = 0 ; Allocate DMA buffer
6460                                    <1>       ;          EAX = Physical address of DMA buffer
6461                                    <1>       ;          ECX = Allocated buffer size in bytes
6462                                    <1>       ;               - page count * 4096 -
6463                                    <1>       ;               (may be bigger than requested)
6464                                    <1>       ;          If BL input > 0,
6465                                    <1>       ;             'sysalloc:' system call will be used with
6466                                    <1>       ;             EBX (for 'sysalloc') = EDX (for 'sysdma')
6467                                    <1>       ;             ECX is same, byte count (buffer size)
6468                                    <1>       ;             EDX = 1024*1024*16 ; 16 MB upper limit
6469                                    <1>       ;          If BL input = 0,
6470                                    <1>       ;             Default DMA buffer (SB16 buffer) will be
6471                                    <1>       ;             checked and if it is free, it's address
6472                                    <1>       ;             will be returned in EAX and it's size
6473                                    <1>       ;             will be returned in ECX (as 65536)
6474                                    <1>       ;
6475                                    <1>       ;          If CF = 1, error code is in EAX
6476                                    <1>       ;             EAX = -1 ; DMA buffer allocation error!
6477                                    <1>       ;              EAX = 11 ; 'Permission Denied' error !
6478                                    <1>       ;
6479                                    <1>       ;              Note: 'sysalloc' error return method
6480                                    <1>       ;                    will be applied if BL input > 0 !
6481                                    <1>       ;
6482                                    <1>       ;       For BH = 1 ; Initialize (Start) DMA
6483                                    <1>       ;           EAX = 0 (Successful)
6484                                    <1>       ;           If CF = 1, error code is in EAX
6485                                    <1>       ;
6486                                    <1>       ;       For BH = 2 ; Get Current DMA Buffer Offset
6487                                    <1>       ;           EAX = DMA Buffer Offset (in bytes)
6488                                    <1>       ;           ;
6489                                    <1>       ;            AX = DMA buffer offset
6490                                    <1>       ;           EAX bits 16 to 23 = Page register value
6491                                    <1>       ;
6492                                    <1>       ;       For BH = 3 ; Get Current DMA count down value
6493                                    <1>       ;           EAX = Count down value (remain bytes)
6494                                    <1>       ;
6495                                    <1>       ;       For BH = 4 ; Get Current DMA channel (in progress)
6496                                    <1>       ;           EAX = DMA channel number (0 to 7)
6497                                    <1>       ;             AH = 0 if the owner is the caller process
6498                                    <1>       ;             AH > 0 if the dma channel is in use by
6499                                    <1>       ;                    another user/process
6500                                    <1>       ;           EAX = -1 (0FFFFFFFFh)
6501                                    <1>       ;                if DMA service is not in use
6502                                    <1>       ;                (stopped or not initialized/started)
6503                                    <1>       ;
6504                                    <1>       ;       For BH = 5 ; Get System's Default DMA Buff Addr
6505                                    <1>       ;           EAX = Default DMA Buffer Address (Physical)
6506                                    <1>       ;             = offset 'sb16_dma_buffer:'
6507                                    <1>       ;           ECX = Buffer size
6508                                    <1>       ;             = 65536
6509                                    <1>       ;
```

```
6510                                    <1>       ;       For BH = 6 ; Get Current DMA Buffer Address
6511                                    <1>       ;           EAX = Current DMA buffer address (Physical)
6512                                    <1>       ;           ECX = Current DMA buffer size (setting value)
6513                                    <1>       ;           Note: These values are for current dma channel
6514                                    <1>       ;                settings for the user/process
6515                                    <1>       ;           ** For now (for current TRDOS 386 version)
6516                                    <1>       ;             only one user/process can use only one
6517                                    <1>       ;             dma channel & one dma buffer at same time
6518                                    <1>       ;             (no multi tasking on DMA service) !!! **
6519                                    <1>       ;           (Once, current DMA user must stop it's own DMA
6520                                    <1>       ;            DMA service, than another user/program
6521                                    <1>       ;            can use DMA service with same dma channel
6522                                    <1>       ;            or with another DMA channel.)
6523                                    <1>       ;
6524                                    <1>       ;       For BH = 7 ; Stop DMA service (for current user
6525                                    <1>       ;           and current DMA channel)
6526                                    <1>       ;           EAX = 0 ; successful
6527                                    <1>       ;           CF = 1 & EAX > 0 (= -1) -> Error
6528                                    <1>
6529 000135EF 80FF07                  <1>       cmp    bh, 7
6530 000135F2 7612                    <1>       jna    short sysdma_0
6531                                    <1>
6532                                    <1> sysdma_err:
6533 000135F4 31C0                    <1>       xor    eax, eax
6534 000135F6 48                      <1>       dec    eax ; -1
6535                                    <1> sysdma_perm_err:
6536 000135F7 A3[64030300]            <1>       mov    [u.r0], eax
6537 000135FC A3[C8030300]            <1>       mov    [u.error], eax ; DMA service error !
6538 00013601 E9AA9DFFFF              <1>       jmp    error
6539                                    <1>
6540                                    <1> sysdma_0:
6541 00013606 08FF                    <1>       or     bh, bh
6542 00013608 0F85BA000000            <1>       jnz    sysdma_1
6543                                    <1>
6544 0001360E 20DB                    <1>       and    bl, bl
6545 00013610 7416                    <1>       jz     short sysdma_01
6546                                    <1>
6547                                    <1>       ; redirect system call to 'sysalloc'
6548 00013612 89D3                    <1>       mov    ebx, edx ; virtual address of DMA buffer
6549                                    <1>       ;ecx = Buffer size in bytes
6550                                    <1>       ; DMA buffer address <= 16MB upper limit
6551 00013614 BA00000001              <1>       mov    edx, 1024*1024*16 ; 16MB limit for DMA buff
6552                                    <1>
6553 00013619 C705[1C940100]FFFF-     <1>       mov    dword [dma_addr], 0FFFFFFFFh ; -1
6553 00013621 FFFF                    <1>
6554                                    <1>
6555 00013623 E9EFE6FFFF              <1>       jmp    sysalloc
6556                                    <1>
6557                                    <1> sysdma_01:
6558 00013628 B8[00000200]            <1>       mov    eax, sb16_dma_buffer
6559                                    <1>
6560 0001362D 803D[D98F0100]01        <1>       cmp    byte [audio_device], 1
6561 00013634 722A                    <1>       jb     short sysdma_03
6562                                    <1>
6563 00013636 3B05[F88F0100]          <1>       cmp    eax, [audio_dma_buff]
6564 0001363C 7507                    <1>       jne    short sysdma_02
6565                                    <1>
6566                                    <1> sysdma_0_err:
6567 0001363E B80B000000              <1>       mov    eax, ERR_PERM_DENIED
6568 00013643 EBB2                    <1>       jmp    short sysdma_perm_err
6569                                    <1>
6570                                    <1> sysdma_02:
6571                                    <1>       ; Only one user is permitted for audio/dma functions
6572                                    <1>
6573 00013645 833D[F88F0100]00        <1>       cmp    dword [audio_dma_buff], 0
6574 0001364C 7612                    <1>       jna    short sysdma_03
6575                                    <1>
6576 0001364E 8A1D[01900100]          <1>       mov    bl, [audio_user]
6577 00013654 08DB                    <1>       or     bl, bl
6578 00013656 7408                    <1>       jz     short sysdma_03
6579                                    <1>
6580 00013658 3A1D[B3030300]          <1>       cmp    bl, [u.uno]
6581 0001365E 75DE                    <1>       jne    short sysdma_0_err
6582                                    <1>
6583                                    <1> sysdma_03:
6584 00013660 8A1D[19940100]          <1>       mov    bl, [dma_user]
6585 00013666 20DB                    <1>       and    bl, bl
6586 00013668 750E                    <1>       jnz    short sysdma_04
6587                                    <1>
6588 0001366A 8A1D[B3030300]          <1>       mov    bl, [u.uno]
6589 00013670 881D[19940100]          <1>       mov    [dma_user], bl
6590                                    <1>
6591 00013676 EB15                    <1>       jmp    short sysdma_05
6592                                    <1>
6593                                    <1> sysdma_04:
6594 00013678 8B35[1C940100]          <1>       mov    esi, [dma_addr]
6595 0001367E 21F6                    <1>       and    esi, esi
6596 00013680 740B                    <1>       jz     short sysdma_05
6597                                    <1>
6598 00013682 46                      <1>       inc    esi ; -1 -> 0
6599 00013683 7408                    <1>       jz     short sysdma_05
6600                                    <1>
6601 00013685 3A1D[B3030300]          <1>       cmp    bl, [u.uno]
6602 0001368B 75B1                    <1>       jne    short sysdma_0_err
6603                                    <1>
6604                                    <1> sysdma_05:
6605                                    <1>       ; edx = virtual address (user's buffer address)
6606                                    <1>       ;
6607 0001368D 81F900000100            <1>       cmp    ecx, 65536   ; byte count (buffer size)
6608 00013693 0F875BFFFFFF            <1>       ja     sysdma_err
6609                                    <1>       ;
6610 00013699 81C1FF0F0000            <1>       add    ecx, PAGE_SIZE-1 ; 4095
6611 0001369F 6681E100F0              <1>       and    cx, ~PAGE_OFF ; not 4095
6612                                    <1>       ;cmp   ecx, 65536
6613                                    <1>       ;ja    sysdma_err ;
```

```
6614 000136A4 51                <1>       push   ecx    ; buffer size (allocated pages * 4096)
6615 000136A5 50                <1>       push   eax    ; offset sb16_dma_buffer
6616 000136A6 89D3              <1>       mov    ebx, edx
6617 000136A8 C1E90C            <1>       shr    ecx, 12 ; byte count to page count
6618                            <1>       ; eax = physical address of (audio) dma buffer
6619                            <1>       ; ebx = virtual address of (audio) dma buffer (user's pgdir)
6620                            <1>       ; ecx = page count (>0)
6621 000136AB E8642BFFFF        <1>       call   direct_memory_access
6622 000136B0 58                <1>       pop    eax
6623 000136B1 59                <1>       pop    ecx
6624 000136B2 0F823CFFFFFF      <1>       jc     sysdma_err
6625                            <1>
6626 000136B8 A3[1C940100]      <1>       mov    [dma_addr], eax
6627 000136BD 890D[20940100]    <1>       mov    [dma_size], ecx ; dma buffer size (in bytes)
6628                            <1>
6629                            <1>       ;mov   [u.r0], eax ; DMA Buffer Address (Physical)
6630                            <1>
6631                            <1>       ;mov   ebp, [u.usp]  ; ebp points to user's registers
6632                            <1>       ;mov   [ebp+24], ecx ; return to user with ecx value
6633                            <1>
6634                            <1>       ;jmp   sysret
6635                            <1>
6636                            <1>       ; 28/08/2017
6637 000136C3 E9C4000000        <1>       jmp    sysdma_51
6638                            <1>
6639                            <1> sysdma_1:
6640 000136C8 80FF01            <1>       cmp    bh, 1
6641 000136CB 0F87A6000000      <1>       ja     sysdma_5
6642                            <1>
6643 000136D1 F6C340            <1>       test   bl, 40h      ; record (read) mode -BL, bit 6-
6644 000136D4 0F851AFFFFFF      <1>       jnz    sysdma_err ; not ready yet!
6645                            <1>
6646 000136DA A1[1C940100]      <1>       mov    eax, [dma_addr] ; physical address of dma buffer
6647 000136DF 21C0              <1>       and    eax, eax
6648 000136E1 0F840DFFFFFF      <1>       jz     sysdma_err
6649                            <1>
6650 000136E7 09D2              <1>       or     edx, edx
6651 000136E9 7504              <1>       jnz    short sysdma_11
6652                            <1>
6653 000136EB 89C2              <1>       mov    edx, eax
6654 000136ED EB08              <1>       jmp    short sysdma_12
6655                            <1> sysdma_11:
6656 000136EF 39C2              <1>       cmp    edx, eax
6657 000136F1 0F82FDFEFFFF      <1>       jb     sysdma_err
6658                            <1> sysdma_12:
6659 000136F7 21C9              <1>       and    ecx, ecx
6660 000136F9 7508              <1>       jnz    short sysdma_13
6661                            <1>
6662 000136FB 8B0D[20940100]    <1>       mov    ecx, [dma_size]
6663 00013701 EB0C              <1>       jmp    short sysdma_14
6664                            <1> sysdma_13:
6665 00013703 3B0D[20940100]    <1>       cmp    ecx, [dma_size]
6666 00013709 0F87E5FEFFFF      <1>       ja     sysdma_err
6667                            <1> sysdma_14:
6668 0001370F 89C6              <1>       mov    esi, eax
6669 00013711 0335[20940100]    <1>       add    esi, [dma_size]
6670                            <1>
6671 00013717 89D0              <1>       mov    eax, edx
6672 00013719 01C8              <1>       add    eax, ecx
6673 0001371B 0F82D3FEFFFF      <1>       jc     sysdma_err ; 02/09/2017
6674                            <1>
6675 00013721 39F0              <1>       cmp    eax, esi
6676 00013723 0F87CBFEFFFF      <1>       ja     sysdma_err
6677                            <1>
6678 00013729 8B3D[F88F0100]    <1>       mov    edi, [audio_dma_buff]
6679 0001372F 8B35[1C940100]    <1>       mov    esi, [dma_addr]
6680                            <1>
6681 00013735 09FF              <1>       or     edi, edi
6682 00013737 7424              <1>       jz     short sysdma_16
6683                            <1>
6684 00013739 803D[D98F0100]01  <1>       cmp    byte [audio_device], 1
6685 00013740 7208              <1>       jb     short sysdma_15
6686                            <1>
6687                            <1>       ; Sound Blaster 16
6688 00013742 39FE              <1>       cmp    esi, edi
6689 00013744 0F84F4FEFFFF      <1>       je     sysdma_0_err ; permmission denied !
6690                            <1>
6691                            <1> sysdma_15:
6692 0001374A C605[1B940100]48  <1>       mov    byte [dma_mode], 48h ; single mode playback
6693                            <1>
6694 00013751 F6C380            <1>       test   bl, 80h ; DMA mode - BL, bit 7, auto init -
6695 00013754 7407              <1>       jz     short sysdma_16
6696                            <1>       ; Auto initialized playback (write) mode
6697 00013756 8005[1B940100]10  <1>       add    byte [dma_mode], 10h ; = 58h
6698                            <1> sysdma_16:
6699 0001375D 80E307            <1>       and    bl, 07h
6700 00013760 881D[1A940100]    <1>       mov    [dma_channel], bl
6701 00013766 8915[24940100]    <1>       mov    [dma_start], edx
6702 0001376C 890D[28940100]    <1>       mov    [dma_count], ecx
6703                            <1>
6704                            <1>       ; 28/08/2017
6705                            <1>       ;call  dma_init
6706                            <1>       ;jmp   sysret
6707 00013772 E94B010000        <1>       jmp    dma_init
6708                            <1>
6709                            <1> sysdma_5:
6710 00013777 80FF05            <1>       cmp    bh, 5
6711 0001377A 7223              <1>       jb     short sysdma_3
6712 0001377C 0F87CE000000      <1>       ja     sysdma_6
6713                            <1>
6714                            <1>       ; Get the system's default dma buffer addr and size
6715 00013782 B8[00000200]      <1>       mov    eax, sb16_dma_buffer
6716 00013787 B900000100        <1>       mov    ecx, 65536 ; Buffer size in bytes
6717                            <1>
6718                            <1> sysdma_51:
```

```
6719                                    <1>       ; 0 = there is not a dma buffer (in use or available)
6720 0001378C A3[64030300]              <1>       mov    [u.r0], eax
6721                                    <1>
6722 00013791 8B2D[60030300]            <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
6723 00013797 894D18                    <1>       mov    [ebp+24], ecx ; return to user with ecx value
6724                                    <1>
6725 0001379A E9319CFFFF                <1>       jmp    sysret
6726                                    <1>
6727                                    <1> sysdma_3:
6728 0001379F 80FF03                    <1>       cmp    bh, 3
6729 000137A2 7231                      <1>       jb     short sysdma_2
6730 000137A4 776B                      <1>       ja     short sysdma_4
6731                                    <1>
6732                                    <1>       ; Get current dma count down value (remain bytes)
6733                                    <1>       ; 28/08/2017
6734 000137A6 0FB635[1A940100]          <1>       movzx  esi, byte [dma_channel]
6735 000137AD 0FB696[A8400100]          <1>       movzx  edx, byte [dma_flip+esi]
6736 000137B4 EE                        <1>       out    dx, al              ; flip-flop clear
6737 000137B5 8A96[88400100]            <1>       mov    dl, [dma_cnt+esi] ; dma count register addr
6738 000137BB EC                        <1>       in     al, dx
6739 000137BC 0FB6D8                    <1>       movzx  ebx, al
6740 000137BF EC                        <1>       in     al, dx
6741 000137C0 88C7                      <1>       mov     bh, al
6742                                    <1>
6743 000137C2 6683FE04                  <1>       cmp    si, 4 ; channel number ?
6744 000137C6 7202                      <1>       jb     short sysdma_31 ; 8 bit dma channel
6745                                    <1>
6746 000137C8 D1E3                      <1>       shl    ebx, 1 ; word count to byte count
6747                                    <1>
6748                                    <1> sysdma_31:
6749 000137CA 891D[64030300]            <1>       mov    [u.r0], ebx
6750                                    <1>
6751 000137D0 E9FB9BFFFF                <1>       jmp    sysret
6752                                    <1>
6753                                    <1> sysdma_2:
6754                                    <1>       ; Get current dma buffer offset (& page)
6755                                    <1>       ; 28/08/2017
6756 000137D5 0FB635[1A940100]          <1>       movzx  esi, byte [dma_channel]
6757 000137DC 0FB696[A8400100]          <1>       movzx  edx, byte [dma_flip+esi]
6758 000137E3 EE                        <1>       out    dx, al              ; flip-flop clear
6759 000137E4 8A96[80400100]            <1>       mov    dl, [dma_adr+esi]
6760 000137EA EC                        <1>       in     al, dx              ; get dma position
6761 000137EB 0FB6D8                    <1>       movzx  ebx, al
6762 000137EE EC                        <1>       in     al, dx
6763 000137EF 88C7                      <1>       mov    bh, al
6764                                    <1>
6765 000137F1 6683FE04                  <1>       cmp    si, 4 ; channel number ?
6766 000137F5 7202                      <1>       jb     short sysdma_21 ; 8 bit dma channel
6767                                    <1>
6768 000137F7 D1E3                      <1>       shl    ebx, 1 ; word offset to byte offset
6769                                    <1>
6770                                    <1> sysdma_21:
6771 000137F9 891D[64030300]            <1>       mov    [u.r0], ebx
6772                                    <1>
6773 000137FF 8A96[90400100]            <1>       mov    dl, [dma_page+esi]
6774 00013805 EC                        <1>       in     al, dx              ; get dma page
6775                                    <1>
6776                                    <1>       ;add   [u.ro+2], al
6777 00013806 0805[66030300]            <1>       or     [u.r0+2], al
6778                                    <1>
6779 0001380C E9BF9BFFFF                <1>       jmp    sysret
6780                                    <1>
6781                                    <1> sysdma_4:
6782                                    <1>       ; Get current DMA channel number
6783                                    <1>       ; 28/08/2017
6784 00013811 8A25[19940100]            <1>       mov    ah, [dma_user]
6785 00013817 20E4                      <1>       and    ah, ah
6786 00013819 750F                      <1>       jnz    short sysdma_42
6787                                    <1>
6788                                    <1> sysdma_41:
6789                                    <1>       ; Not a valid dma channel (in use)
6790 0001381B C705[64030300]FFFF-       <1>       mov    dword [u.r0], -1 ; 0FFFFFFFFh
6790 00013823 FFFF                      <1>
6791 00013825 E9A69BFFFF                <1>       jmp    sysret
6792                                    <1>
6793                                    <1> sysdma_42:
6794 0001382A 8B35[1C940100]            <1>       mov    esi, [dma_addr]
6795 00013830 21F6                      <1>       and    esi, esi
6796 00013832 74E7                      <1>       jz     short sysdma_41
6797                                    <1>
6798 00013834 46                        <1>       inc    esi ; -1 -> 0
6799 00013835 74E4                      <1>       jz     short sysdma_41
6800                                    <1>
6801 00013837 A0[1A940100]              <1>       mov    al, [dma_channel]
6802                                    <1>
6803 0001383C 3A25[B3030300]            <1>       cmp    ah, [u.uno]
6804 00013842 7502                      <1>       jne    short sysdma_43
6805                                    <1>
6806 00013844 30E4                      <1>       xor    ah, ah ; DMA channel in use by current user
6807                                    <1>
6808                                    <1> sysdma_43:
6809 00013846 A3[64030300]              <1>       mov    [u.r0], eax ; AL = dma channel number
6810                                    <1>                          ; AH > 0 if the the channel
6811                                    <1>                          ; in use by another user/process
6812 0001384B E9809BFFFF                <1>       jmp    sysret
6813                                    <1>
6814                                    <1> sysdma_6:
6815 00013850 80FF06                    <1>       cmp    bh, 6
6816 00013853 7710                      <1>       ja     short sysdma_7
6817                                    <1>
6818                                    <1>       ; 28/08/2017
6819                                    <1>       ; Get current DMA buffer addr and size
6820 00013855 A1[1C940100]              <1>       mov    eax, [dma_addr] ; dma buffer address
6821 0001385A 8B0D[20940100]            <1>       mov    ecx, [dma_size] ; dma buffer size (in bytes)
6822                                    <1>
```

```
6823 00013860 E927FFFFFF          <1>       jmp    sysdma_51
6824                              <1>
6825                              <1> sysdma_7:
6826                              <1>       ; DMA service STOP
6827 00013865 A0[B3030300]        <1>       mov    al, [u.uno]
6828 0001386A 3A05[19940100]      <1>       cmp    al, [dma_user]
6829 00013870 751D                <1>       jne    short sysdma_72
6830                              <1>
6831 00013872 28C0                <1>       sub    al, al ; 0
6832                              <1>
6833 00013874 A2[19940100]        <1>       mov    [dma_user], al ; clear user
6834                              <1>
6835 00013879 8605[1B940100]      <1>       xchg   al, [dma_mode]
6836 0001387F 20C0                <1>       and    al, al
6837                              <1>       ;jz    short sysdma_err
6838 00013881 7527                <1>       jnz    short sysdma_73
6839                              <1>
6840                              <1> sysdma_71:
6841 00013883 31C0                <1>       xor    eax, eax
6842 00013885 A3[64030300]        <1>       mov    [u.r0], eax; 0
6843 0001388A E9419BFFFF          <1>       jmp    sysret
6844                              <1>
6845                              <1> sysdma_72:
6846                              <1>       ; 28/08/2017
6847 0001388F 803D[19940100]00    <1>       cmp    byte [dma_user], 0
6848 00013896 76EB                <1>       jna    short sysdma_71 ; Nothing to do !
6849                              <1>
6850 00013898 833D[1C940100]00    <1>       cmp    dword [dma_addr], 0
6851 0001389F 0F8799FDFFFF        <1>       ja     sysdma_0_err
6852                              <1>
6853 000138A5 A2[19940100]        <1>       mov    [dma_user], al ; reset to current user
6854                              <1>
6855                              <1> sysdma_73:
6856                              <1>       ; 28/08/2017
6857 000138AA 0FB635[1A940100]    <1>       movzx  esi, byte [dma_channel]
6858 000138B1 0FB696[98400100]    <1>       movzx  edx, byte [dma_mask+esi]
6859 000138B8 A0[1A940100]        <1>       mov    al, [dma_channel]
6860 000138BD 0C04                <1>       or     al, 4
6861 000138BF EE                  <1>       out    dx, al
6862                              <1>
6863 000138C0 EBC1                <1>       jmp    short sysdma_71
6864                              <1>
6865                              <1> dma_init:
6866                              <1>       ; 28/08/2017
6867                              <1>       ; 20/08/2017
6868                              <1>       ; DMA initialization
6869                              <1>       ; 14/08/2017
6870                              <1>       ; 03/08/2017, 06/08/2017, 08/08/2017
6871                              <1>       ; 02/07/2017, 13/07/2017, 16/07/2017, 30/07/2017
6872                              <1>       ; (Derived from 'DMA_INIT' procedure in SB16MOD.ASM)
6873                              <1>       ; Modified for TRDOS 386 DMA buffer allocation & initialization !
6874                              <1>
6875 000138C2 8B1D[24940100]     <1>       mov    ebx, [dma_start]
6876 000138C8 8B0D[28940100]     <1>       mov    ecx, [dma_count]
6877                              <1>
6878 000138CE 0FB635[1A940100]    <1>       movzx  esi, byte [dma_channel]
6879                              <1>
6880 000138D5 6683FE04            <1>       cmp    si, 4
6881 000138D9 7205                <1>       jb     short gdmi1
6882                              <1>       ; 08/08/2017
6883 000138DB 66D1E9              <1>       shr    cx, 1 ; word count
6884 000138DE D1EB                <1>       shr    ebx, 1 ; convert byte offset to word offset
6885                              <1> gdmi1:
6886                              <1>       ;mov   [dma_poff], bx ; 08/08/2017
6887 000138E0 6649                <1>       dec    cx                  ; dma size = block size - 1
6888                              <1>
6889 000138E2 0FB696[98400100]    <1>       movzx  edx, byte [dma_mask+esi] ; 30/07/2017
6890 000138E9 A0[1A940100]        <1>       mov    al, [dma_channel]
6891 000138EE 0C04                <1>       or     al, 4
6892 000138F0 EE                  <1>       out    dx, al              ; dma channel mask
6893                              <1>
6894 000138F1 30C0                <1>       xor    al, al ; 0 ; any value ! 08/08/2017
6895 000138F3 8A96[A8400100]      <1>       mov    dl, [dma_flip+esi]
6896 000138F9 EE                  <1>       out    dx, al              ; flip-flop clear
6897                              <1>
6898 000138FA 8A96[A0400100]      <1>       mov    dl, [dma_mod+esi]
6899 00013900 A0[1A940100]        <1>       mov    al, [dma_channel]  ; 13/07/2017
6900 00013905 2403                <1>       and    al, 3
6901                              <1>       ; 08/08/2017
6902 00013907 0A05[1B940100]      <1>       or     al, [dma_mode] ; 58h    ; dma mode for SB16
6903 0001390D EE                  <1>       out    dx, al
6904                              <1>
6905 0001390E 8A96[80400100]      <1>       mov    dl, [dma_adr+esi]
6906 00013914 88D8                <1>       mov    al, bl
6907 00013916 EE                  <1>       out    dx, al              ; offset low
6908                              <1>
6909 00013917 88F8                <1>       mov    al, bh
6910 00013919 EE                  <1>       out    dx, al              ; offset high
6911                              <1>
6912 0001391A 8A96[88400100]      <1>       mov    dl, [dma_cnt+esi]
6913 00013920 88C8                <1>       mov    al, cl
6914 00013922 EE                  <1>       out    dx, al              ; size low
6915                              <1>
6916 00013923 88E8                <1>       mov    al, ch
6917 00013925 EE                  <1>       out    dx, al              ; size high
6918                              <1>
6919 00013926 8A96[90400100]      <1>       mov    dl, [dma_page+esi]
6920                              <1>       ; 14/08/2017
6921 0001392C 6683FE04            <1>       cmp    si, 4
6922 00013930 7305                <1>       jnb    short gdmi2
6923 00013932 C1EB10              <1>       shr    ebx, 16
6924 00013935 EB06                <1>       jmp    short gdmi3
6925                              <1> gdmi2:
6926                              <1>       ; 09/08/2017
6927 00013937 C1EB0F              <1>       shr    ebx, 15       ; complete 16 bit shift
```

```
6928 0001393A 80E3FE              <1>       and    bl, 0FEh ; clear bit 0 (not necessary)
6929                              <1> gdmi3:
6930 0001393D 88D8                <1>       mov    al, bl
6931 0001393F EE                  <1>       out    dx, al              ; page
6932                              <1>
6933 00013940 8A96[98400100]      <1>       mov    dl, [dma_mask+esi]
6934 00013946 A0[1A940100]        <1>       mov    al, [dma_channel]  ; 13/07/2017
6935 0001394B 2403                <1>       and    al, 3
6936 0001394D EE                  <1>       out    dx, al              ; dma channel unmask
6937                              <1>
6938                              <1>       ;retn
6939                              <1>       ; 28/08/2017
6940 0001394E E97D9AFFFF          <1>       jmp    sysret
6941                              <1>
6942                              <1> otty:
6943                              <1> sret:
6944                              <1> ocvt:
6945                              <1> ctty:
6946                              <1> cret:
6947                              <1> ccvt:
6948                              <1> rtty:
6949                              <1> wtty:
6950                              <1> rmem:
6951                              <1> wmem:
6952                              <1> rfd:
6953                              <1> rhd:
6954                              <1> wfd:
6955                              <1> whd:
6956                              <1> rlpt:
6957                              <1> wlpt:
6958                              <1> rcvt:
6959                              <1> xmtt:
6960 00013953 C3                  <1>       retn
```

```
3114                                  %include 'trdosk9.s' ; 04/01/2016
3115                              <1> ; ****************************************************************************
3116                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.4) - INITIALIZED DATA : trdosk9.s
3117                              <1> ; ----------------------------------------------------------------------------
3118                              <1> ; Last Update: 18/04/2021
3119                              <1> ; ----------------------------------------------------------------------------
3120                              <1> ; Beginning: 04/01/2016
3121                              <1> ; ----------------------------------------------------------------------------
3122                              <1> ; ----------------------------------------------------------------------------
3123                              <1> ; Assembler: NASM version 2.15 (trdos386.s)
3124                              <1> ; ----------------------------------------------------------------------------
3125                              <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
3126                              <1> ; TRDOS2.ASM (09/11/2011)
3127                              <1> ; ****************************************************************************
3128                              <1> ; DRV_INIT.ASM [26/09/2009] Last Update: 07/08/2011
3129                              <1> ; MAINPROG.ASM [17/01/2004] Last Update: 09/11/2011
3130                              <1> ; CMD_INTR.ASM [29/01/2005] Last Update: 09/11/2011
3131                              <1> ; FILE.ASM [29/10/2009] Last Update: 09/10/2011
3132                              <1>
3133                              <1> ; 12/02/2016
3134                              <1> Last_DOS_DiskNo:
3135 00013954 01                 <1>             db 1 ; A: = 0 & B: = 1
3136                              <1>
3137                              <1> Restore_CDIR:
3138 00013955 FF                  <1>             db 0FFh ; Initial value -> any number except 0
3139                              <1>
3140                              <1> msg_CRLF_temp:
3141 00013956 070D0A00            <1>             db 07h, 0Dh, 0Ah, 0
3142                              <1>
3143                              <1> Magic_Bytes:
3144 0001395A 04                  <1>             db 4
3145 0001395B 01                  <1>             db 1
3146                              <1> mainprog_Version:
3147 0001395C 07                  <1>             db 7
3148 0001395D 5B5452444F535D204D- <1>             db "[TRDOS] Main Program v2.0.180421"
3148 00013966 61696E2050726F6772- <1>
3148 0001396F 616D2076322E302E31- <1>
3148 00013978 3830343231          <1>
3149 0001397D 0D0A                <1>             db 0Dh, 0Ah
3150 0001397F 286329204572646F67- <1>             db "(c) Erdogan Tan 2005-2021"
3150 00013988 616E2054616E203230- <1>
3150 00013991 30352D32303231      <1>
3151 00013998 0D0A00              <1>             db 0Dh, 0Ah, 0
3152                              <1>
3153                              <1> MainProgCfgFile: ; 14/04/2016
3154 0001399B 4D41494E50524F472E- <1>             db "MAINPROG.CFG", 0
3154 000139A4 43464700            <1>
3155                              <1>
3156                              <1> TRDOSPromptLabel:
3157 000139A8 5452444F53          <1>             db "TRDOS"
3158 000139AD 00                  <1>             db 0
3159 000139AE 00<rep 5h>          <1>                times 5 db 0
3160 000139B3 00                  <1>             db 0
3161                              <1>
3162                              <1> ; INTERNAL COMMANDS
3163                              <1> Command_List:
3164 000139B4 44495200            <1> Cmd_Dir:    db "DIR", 0
3165 000139B8 434400              <1> Cmd_Cd:     db "CD", 0
3166 000139BB 433A00              <1> Cmd_Drive:  db "C:", 0
3167 000139BE 56455200            <1> Cmd_Ver:    db "VER", 0
3168 000139C2 4558495400          <1> Cmd_Exit:   db "EXIT", 0
3169 000139C7 50524F4D505400      <1> Cmd_Prompt: db "PROMPT", 0
3170 000139CE 564F4C554D4500      <1> Cmd_Volume: db "VOLUME", 0
3171 000139D5 4C4F4E474E414D4500  <1> Cmd_LongName: db "LONGNAME", 0
3172 000139DE 4441544500          <1> Cmd_Date:   db "DATE", 0
3173 000139E3 54494D4500          <1> Cmd_Time:   db "TIME", 0
3174 000139E8 52554E00            <1> Cmd_Run:    db "RUN", 0
3175 000139EC 53455400            <1> Cmd_Set:    db "SET", 0
3176 000139F0 434C5300            <1> Cmd_Cls:    db "CLS", 0
3177 000139F4 53484F5700          <1> Cmd_Show:   db "SHOW", 0
3178 000139F9 44454C00            <1> Cmd_Del:    db "DEL", 0
3179 000139FD 41545452494200      <1> Cmd_Attrib: db "ATTRIB", 0
```

```
3180 00013A04 52454E414D4500      <1> Cmd_Rename:  db "RENAME", 0
3181 00013A0B 524D44495200        <1> Cmd_Rmdir:   db "RMDIR", 0
3182 00013A11 4D4B44495200        <1> Cmd_Mkdir:   db "MKDIR", 0
3183 00013A17 434F505900          <1> Cmd_Copy:    db "COPY", 0
3184 00013A1C 4D4F564500          <1> Cmd_Move:    db "MOVE", 0
3185 00013A21 5041544800          <1> Cmd_Path:    db "PATH", 0
3186 00013A26 4D454D00            <1> Cmd_Mem:     db "MEM", 0
3187 00013A2A 00                  <1>              db 0
3188 00013A2B 46494E4400          <1> Cmd_Find:    db "FIND", 0
3189 00013A30 4543484F00          <1> Cmd_Echo:    db "ECHO", 0
3190 00013A35 2A00                <1> Cmd_Remark:  db "*", 0
3191 00013A37 3F00                <1> Cmd_Help:    db "?", 0
3192 00013A39 44455649434500      <1> Cmd_Device:  db "DEVICE", 0
3193 00013A40 4445564C49535400    <1> Cmd_DevList: db "DEVLIST", 0
3194 00013A48 434844495200        <1> Cmd_Chdir:   db "CHDIR", 0
3195 00013A4E 4245455000          <1> Cmd_Beep:    db "BEEP", 0
3196                              <1>
3197 00013A53 00                  <1>              db 0
3198                              <1>
3199                              <1> ; 15/02/2016 (FILE.ASM, 09/10/2011)
3200                              <1> invalid_fname_chars:
3201 00013A54 222728292A2B2C2F    <1>              db 22h, 27h, 28h, 29h, 2Ah, 2Bh, 2Ch, 2Fh
3202 00013A5C 3A3B3C3D3E3F40      <1>              db 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh, 40h
3203 00013A63 5B5C5D5E60          <1>              db 5Bh, 5Ch, 5Dh, 5Eh, 60h
3204                              <1> sizeInvFnChars  equ ($ - invalid_fname_chars)
3205                              <1> ;
3206                              <1>
3207                              <1> Msg_Enter_Date:
3208 00013A68 456E746572206E6577- <1>                 db 'Enter new date (dd-mm-yy): '
3208 00013A71 206461746520286464- <1>
3208 00013A7A 2D6D6D2D7979293A20  <1>
3209 00013A83 00                  <1>                 db 0
3210                              <1> Msg_Show_Date:
3211 00013A84 43757272656E742064- <1>                 db   'Current date is '
3211 00013A8D 61746520697320      <1>
3212 00013A94 30                  <1> Day:            db   '0'
3213 00013A95 30                  <1>                 db   '0'
3214 00013A96 2F                  <1>                 db   '/'
3215 00013A97 30                  <1> Month:          db   '0'
3216 00013A98 30                  <1>                 db   '0'
3217 00013A99 2F                  <1>                 db   '/'
3218 00013A9A 30                  <1> Century:        db   '0'
3219 00013A9B 30                  <1>                 db   '0'
3220 00013A9C 30                  <1> Year:           db   '0'
3221 00013A9D 30                  <1>                 db   '0'
3222 00013A9E 0D0A00              <1>                 db   0Dh, 0Ah, 0
3223                              <1>
3224                              <1> Msg_Enter_Time:
3225 00013AA1 456E746572206E6577- <1>                 db 'Enter new time: '
3225 00013AAA 2074696D653A20      <1>
3226 00013AB1 00                  <1>                 db 0
3227                              <1> Msg_Show_Time:
3228 00013AB2 43757272656E742074- <1>                 db   'Current time is '
3228 00013ABB 696D6520697320      <1>
3229 00013AC2 30                  <1> Hour:           db   '0'
3230 00013AC3 30                  <1>                 db   '0'
3231 00013AC4 3A                  <1>                 db   ':'
3232 00013AC5 30                  <1> Minute:         db   '0'
3233 00013AC6 30                  <1>                 db   '0'
3234 00013AC7 3A                  <1>                 db   ':'
3235 00013AC8 30                  <1> Second:         db   '0'
3236 00013AC9 30                  <1>                 db   '0'
3237 00013ACA 0D0A00              <1>                 db   0Dh, 0Ah, 0
3238                              <1>
3239                              <1> ;VolSize_Unit1:   dd 0
3240                              <1> ;VolSize_Unit2:   dd 0
3241                              <1>
3242                              <1> VolSize_KiloBytes:
3243 00013ACD 206B696C6F62797465- <1>                 db " kilobytes", 0Dh, 0Ah, 0
3243 00013AD6 730D0A00            <1>
3244                              <1> VolSize_Bytes:
3245 00013ADA 2062797465730D0A00  <1>                 db " bytes", 0Dh, 0Ah, 0
3246                              <1> Volume_in_drive:
3247 00013AE3 0D0A                <1>                 db 0Dh, 0Ah
3248                              <1> Vol_FS_Name:
3249 00013AE5 54522046533120      <1>                 db "TR FS1 "
3250 00013AEC 566F6C756D6520696E- <1>                 db "Volume in drive "
3250 00013AF5 20647269766520      <1>
3251 00013AFC 30                  <1> Vol_Drv_Name:   db 30h
3252 00013AFD 3A                  <1>                 db ":"
3253 00013AFE 20697320            <1>                 db " is "
3254 00013B02 0D0A00              <1>                 db 0Dh, 0Ah, 0
3255                              <1> Dir_Drive_Str:
3256 00013B05 54522D444F53204472- <1>                  db "TR-DOS Drive "
3256 00013B0E 69766520            <1>
3257                              <1> Dir_Drive_Name:
3258 00013B12 303A                <1>                  db "0:"
3259 00013B14 0D0A                <1>                  db  0Dh, 0Ah
3260                              <1> Vol_Str_Header:
3261 00013B16 566F6C756D65204E61- <1>                  db "Volume Name: "
3261 00013B1F 6D653A20            <1>
3262                              <1> Vol_Name:
3263 00013B23 00<rep 40h>         <1>                 times 64 db 0
3264 00013B63 00                  <1>                 db 0
3265                              <1> Vol_Serial_Header:
3266 00013B64 0D0A                <1>                 db 0Dh, 0Ah
3267 00013B66 566F6C756D65205365- <1>                 db "Volume Serial No: "
3267 00013B6F 7269616C204E6F3A20  <1>
3268                              <1> Vol_Serial1:
3269 00013B78 30303030            <1>                 db "0000"
3270 00013B7C 2D                  <1>                 db "-"
3271                              <1> Vol_Serial2:
3272 00013B7D 30303030            <1>                 db "0000"
3273 00013B81 0D0A00              <1>                 db 0Dh, 0Ah, 0
3274                              <1>
```

```
                            <1> ;Vol_Tot_Sec_Str_Start:
                            <1> ;           dd 0
                            <1> Vol_Total_Sector_Header:
00013B84 0D0A               <1>             db 0Dh, 0Ah
00013B86 566F6C756D65205369- <1>            db "Volume Size : ", 0
00013B8F 7A65203A2000       <1>
                            <1> ;Vol_Tot_Sec_Str:
                            <1> ;           db "0000000000"
                            <1> ;Vol_Tot_Sec_Str_End:
                            <1> ;           db 0
                            <1> ;Vol_Free_Sectors_Str_Start:
                            <1> ;           dd 0
                            <1> Vol_Free_Sectors_Header:
00013B95 467265652053706163- <1>            db "Free Space  : ", 0
00013B9E 6520203A2000       <1>
                            <1> ;Vol_Free_Sectors_Str:
                            <1> ;           db "0000000000"
                            <1> ;Vol_Free_Sectors_Str_End:
                            <1> ;           db 0
                            <1>
                            <1> Dir_Str_Header:
00013BA4 4469726563746F7279- <1>                db "Directory: "
00013BAD 3A20               <1>
00013BAF 2F                 <1> Dir_Str_Root:   db "/"
00013BB0 00<rep 40h>        <1> Dir_Str:        times 64 db 0
00013BF0 00000000           <1>                 dd 0
00013BF4 00                 <1>                 db 0
                            <1>
                            <1> Msg_Bad_Command:
00013BF5 42616420636F6D6D61- <1>                db "Bad command or file name!"
00013BFE 6E64206F722066696C- <1>
00013C07 65206E616D6521     <1>
00013C0E 0D0A00             <1>                 db 0Dh, 0Ah, 0
                            <1>
                            <1> msgl_drv_not_ready:
00013C11 070D0A             <1>             db 07h, 0Dh, 0Ah
                            <1>
                            <1> ; CMD_INTR.ASM - 09/11/2011 - Messages
                            <1>
                            <1> Msg_Not_Ready_Read_Err:
00013C14 4472697665206E6F74- <1>                db "Drive not ready or read error!"
00013C1D 207265616479206F72- <1>
00013C26 207265616420657272- <1>
00013C2F 6F7221             <1>
00013C32 0D0A00             <1>                 db 0Dh, 0Ah, 0
                            <1>
                            <1> Msg_Not_Ready_Write_Err:
00013C35 4472697665206E6F74- <1>                db "Drive not ready or write error!"
00013C3E 207265616479206F72- <1>
00013C47 207772697465206572- <1>
00013C50 726F7221           <1>
00013C54 0D0A00             <1>                 db 0Dh, 0Ah, 0
                            <1>
                            <1> Msg_Dir_Not_Found:
00013C57 4469726563746F7279- <1>                db "Directory not found!"
00013C60 206E6F7420666F756E- <1>
00013C69 6421               <1>
00013C6B 0D0A00             <1>                 db 0Dh, 0Ah, 0
                            <1>
                            <1> Msg_File_Not_Found:
00013C6E 46696C65206E6F7420- <1>                db "File not found!"
00013C77 666F756E6421       <1>
00013C7D 0D0A00             <1>                 db 0Dh, 0Ah, 0
                            <1>
                            <1> Msg_File_Directory_Not_Found:
00013C80 46696C65206F722064- <1>                db "File or directory not found!"
00013C89 69726563746F727920- <1>
00013C92 6E6F7420666F756E64- <1>
00013C9B 21                 <1>
00013C9C 0D0A00             <1>                 db 0Dh, 0Ah, 0
                            <1>
                            <1> Msg_LongName_Not_Found:
00013C9F 4C6F6E67206E616D65- <1>                db "Long name not found!"
00013CA8 206E6F7420666F756E- <1>
00013CB1 6421               <1>
00013CB3 0D0A00             <1>                 db 0Dh, 0Ah, 0
                            <1>
                            <1> beep_Insufficient_Memory: ; 20/02/2017
00013CB6 0D0A               <1>             db 0Dh, 0Ah
00013CB8 07                 <1>             db 07h
                            <1> Msg_Insufficient_Memory:
00013CB9 496E73756666696369- <1>                db "Insufficient memory!"
00013CC2 656E74206D656D6F72- <1>
00013CCB 7921               <1>
00013CCD 0D0A00             <1>                 db 0Dh, 0Ah, 0
                            <1>
                            <1> Msg_Error_Code:
00013CD0 436F6D6D616E642066- <1>                db 'Command failed! Error code : '
00013CD9 61696C656421204572- <1>
00013CE2 726F7220636F646520- <1>
00013CEB 3A20               <1>
00013CED 303068             <1> error_code_hex: db '00h'
00013CF0 0A0A00             <1>                 db 0Ah, 0Ah, 0
                            <1>
00013CF3 90                 <1> align 2
                            <1>
                            <1> ; 10/02/2016
                            <1> ; DIR.ASM - 09/10/2011
                            <1>
00013CF4 3C4449523E20202020- <1> Type_Dir:    db '<DIR>     ' ; 10 bytes
00013CFD 20                 <1>
                            <1>
                            <1> File_Name:
00013CFE 20<rep Ch>         <1>             times 12 db 20h
00013D0A 20                 <1>             db 20h
```

```
                                <1> Dir_Or_FileSize:
00013D0B 20<rep Ah>             <1>             times 10 db 20h
00013D15 20                     <1>             db 20h
                                <1> File_Attribute:
00013D16 20202020               <1>             dd 20202020h
00013D1A 20                     <1>             db 20h
                                <1> File_Day:
00013D1B 3030                   <1>              db '0','0'
00013D1D 2F                     <1>             db '/'
                                <1> File_Month:
00013D1E 3030                   <1>             db '0','0'
00013D20 2F                     <1>             db '/'
                                <1> File_Year:
00013D21 30303030               <1>             db '0','0','0','0'
00013D25 20                     <1>             db 20h
                                <1> File_Hour:
00013D26 3030                   <1>             db '0','0'
00013D28 3A                     <1>             db ':'
                                <1> File_Minute:
00013D29 3030                   <1>             db '0','0'
00013D2B 00                     <1>             db 0
                                <1>
                                <1> Decimal_File_Count_Header:
00013D2C 0D0A                   <1>             db 0Dh, 0Ah
                                <1> Decimal_File_Count:
00013D2E 00<rep 6h>             <1>             times 6 db 0
                                <1>
00013D34 2066696C6528732920-    <1> str_files:  db " file(s) & "
00013D3D 2620                   <1>
                                <1> Decimal_Dir_Count:
00013D3F 00<rep 6h>             <1>             times 6 db 0
                                <1> str_dirs:
00013D45 206469726563746F72-    <1>             db " directory(s) "
00013D4E 7928732920             <1>
00013D53 0D0A00                 <1>             db 0Dh, 0Ah, 0
                                <1>
00013D56 206279746528732920-    <1> str_bytes:  db " byte(s) in file(s)"
00013D5F 696E2066696C652873-    <1>
00013D68 29                     <1>
00013D69 0D0A00                 <1>             db 0Dh, 0Ah, 0
                                <1>
                                <1> ; CMD_INTR.ASM - 09/11/2011
                                <1> ; 07/10/2010
                                <1> Msg_invalid_name_chars:
00013D6C 496E76616C69642066-    <1>             db "Invalid file or directory name characters!"
00013D75 696C65206F72206469-    <1>
00013D7E 726563746F7279206E-    <1>
00013D87 616D65206368617261-    <1>
00013D90 637465727321           <1>
00013D96 0D0A00                 <1>             db 0Dh, 0Ah, 0
                                <1> ; 21/02/2016
00013D99 46696C65206F722064-    <1> Msg_Name_Exists: db "File or directory name exists!"
00013DA2 69726563746F727920-    <1>
00013DAB 6E616D652065786973-    <1>
00013DB4 747321                 <1>
00013DB7 0D0A00                 <1>                 db 0Dh, 0Ah, 0
                                <1> Msg_DoYouWantMkdir:
00013DBA 446F20796F75207761-    <1>                 db "Do you want to make directory ", 0
00013DC3 6E7420746F206D616B-    <1>
00013DCC 652064696726563746F-    <1>
00013DD5 72792000               <1>
00013DD9 2028592F4E29203F20-    <1> Msg_YesNo:      db " (Y/N) ? ", 0
00013DE2 00                     <1>
00013DE3 000D0A00               <1> Y_N_nextline:   db 0, 0Dh, 0Ah, 0
00013DE7 4F4B2E0D0A00           <1> Msg_OK:         db "OK.", 0Dh, 0Ah, 0
                                <1>
                                <1> ; 27/02/2016
                                <1> Msg_DoYouWantRmDir:
00013DED 446F20796F75207761-    <1>                 db "Do you want to delete directory ", 0
00013DF6 6E7420746F2064656C-    <1>
00013DFF 657465206469726563-    <1>
00013E08 746F72792000           <1>
                                <1> Msg_Dir_Not_Empty:
00013E0E 4469726563746F7279-    <1>                 db "Directory not empty!"
00013E17 206E6F7420656D7074-    <1>
00013E20 7921                   <1>
00013E22 0D0A00                 <1>                 db 0Dh, 0Ah, 0
                                <1>
                                <1> Msg_DoYouWantDelete:
00013E25 446F20796F75207761-    <1>                 db "Do you want to delete file ",0
00013E2E 6E7420746F2064656C-    <1>
00013E37 6574652066696C6520-    <1>
00013E40 00                     <1>
                                <1>
00013E41 44656C657465642E2E-    <1> Msg_Deleted:    db "Deleted...", 0Dh, 0Ah, 0
00013E4A 2E0D0A00               <1>
                                <1>
                                <1> Msg_Permission_Denied:
00013E4E 07                     <1>                 db 7
00013E4F 5065726D697373696F-    <1>                 db "Permission denied!", 0Dh, 0Ah, 0
00013E58 6E2064656E69656421-    <1>
00013E61 0D0A00                 <1>
                                <1>
                                <1> ; 04/03/2016
00013E64 4E657720               <1> Msg_New:        db "New "
00013E68 00                     <1>                 db 0
                                <1> Str_Attributes:
00013E69 417474726962757465-    <1>                 db "Attributes : "
00013E72 73203A20               <1>
00013E76 4E4F524D414C           <1> Attr_Chars:     db "NORMAL"
00013E7C 00                     <1>                 db 0
                                <1>
                                <1> ; 06/03/2016
                                <1> ; CMD_INTR.ASM - 16/11/2010
                                <1> Msg_DoYouWantRename:
```

```
3433 00013E7D 446F20796F75207761- <1>             db "Do you want to rename ", 0
3433 00013E86 6E7420746F2072656E- <1>
3433 00013E8F 616D652000          <1>
3434 00013E94 66696C652000        <1> Rename_File: db "file ", 0
3435 00013E9A 6469726563746F7279- <1> Rename_Directory: db "directory ", 0
3435 00013EA3 2000                <1>
3436 00013EA5 00<rep Dh>          <1> Rename_OldName: times 13 db 0
3437 00013EB2 20617320            <1> Msg_File_rename_as: db " as "
3438 00013EB6 00<rep Dh>          <1> Rename_NewName: times 13 db 0
3439                               <1>
3440                               <1> ; 08/03/2016
3441                               <1> ; CMD_INTR.ASM - 01/08/2010 - 23/04/2011
3442                               <1> msg_not_same_drv:
3443 00013EC3 4E6F742073616D6520- <1>             db "Not same drive!"
3443 00013ECC 647269766521        <1>
3444 00013ED2 0D0A00              <1>             db 0Dh, 0Ah, 0
3445                               <1>
3446                               <1> Msg_DoYouWantMoveFile:
3447 00013ED5 446F20796F75207761- <1>             db "Do you want to move file", 0
3447 00013EDE 6E7420746F206D6F76- <1>
3447 00013EE7 652066696C6500      <1>
3448                               <1>
3449                               <1> msg_insufficient_disk_space:
3450 00013EEE 496E73756666696369- <1>             db "Insufficient disk space!"
3450 00013EF7 656E742046469736B20- <1>
3450 00013F00 737061636521        <1>
3451 00013F06 0D0A00              <1>             db 0Dh, 0Ah, 0
3452                               <1>
3453                               <1> ; 01/08/2010
3454                               <1> msg_source_file:
3455 00013F09 0D0A536F7572636520- <1>             db 0Dh, 0Ah, "Source file name      :   "
3455 00013F12 66696C65206E616D65- <1>
3455 00013F1B 2020202020203A2020- <1>
3455 00013F24 20                  <1>
3456                               <1> msg_source_file_drv:
3457 00013F25 203A00              <1>             db " :", 0
3458                               <1> msg_destination_file:
3459 00013F28 0D0A44657374696E61- <1>             db 0Dh, 0Ah, "Destination file name :   "
3459 00013F31 74696F6E2066696C65- <1>
3459 00013F3A 206E616D65203A2020- <1>
3459 00013F43 20                  <1>
3460                               <1> msg_destination_file_drv:
3461 00013F44 203A00              <1>             db " :", 0
3462                               <1> msg_copy_nextline:
3463 00013F47 0D0A00              <1>             db 0Dh, 0Ah, 0
3464                               <1>
3465                               <1> ; 15/03/2016
3466                               <1> ; CMD_INTR.ASM
3467                               <1>
3468                               <1> Msg_DoYouWantOverWriteFile:
3469 00013F4A 446F20796F75207761- <1>             db "Do you want to overwrite file ",0
3469 00013F53 6E7420746F206F7665- <1>
3469 00013F5C 727772697465206669- <1>
3469 00013F65 6C652000            <1>
3470                               <1>
3471                               <1> Msg_DoYouWantCopyFile:
3472 00013F69 446F20796F75207761- <1>             db "Do you want to copy file",0
3472 00013F72 6E7420746F20636F70- <1>
3472 00013F7B 792066696C6500      <1>
3473                               <1>
3474                               <1> Msg_read_file_error_before_EOF:
3475 00013F82 46696C652072656164- <1>             db "File reading error! (before EOF)"
3475 00013F8B 696E67206572726F72- <1>
3475 00013F94 2120286265666F7265- <1>
3475 00013F9D 20454F4629          <1>
3476 00013FA2 0A0A00              <1>             db 0Ah, 0Ah, 0
3477                               <1>
3478                               <1> ; 18/03/2016
3479                               <1> ; TRDOS 386 (v2.0) mainprog copy procedure
3480                               <1> msg_reading:
3481 00013FA5 52656164696E672E2E- <1>             db "Reading... ", 0
3481 00013FAE 2E2000              <1>
3482                               <1> msg_writing:
3483 00013FB1 57726974696E672E2E- <1>             db "Writing... ", 0
3483 00013FBA 2E2000              <1>
3484                               <1> percentagestr:
3485 00013FBD 2020202500          <1>             db "   %", 0  ; "  0%" .. "100%"
3486                               <1> ; 11/04/2016
3487                               <1> Msg_No_Set_Space:
3488 00013FC2 496E73756666696369- <1>             db "Insufficient environment space!"
3488 00013FCB 656E7420656E766972- <1>
3488 00013FD4 6F6E6D656E74207370- <1>
3488 00013FDD 61636521            <1>
3489 00013FE1 0D0A00              <1>             db 0Dh, 0Ah, 0
3490                               <1> ; 18/04/2016
3491                               <1> isc_msg:
3492 00013FE4 0D0A                <1>             db 0Dh, 0Ah
3493 00013FE6 494E56414C49442053- <1>             db "INVALID SYSTEM CALL", 0
3493 00013FEF 595354454D2043414C- <1>
3493 00013FF8 4C00                <1>
3494                               <1> usi_msg:
3495 00013FFA 0D0A                <1>             db 0Dh, 0Ah
3496 00013FFC 554E444546494E4544- <1>             db "UNDEFINED SOFTWARE INTERRUPT", 0
3496 00014005 20534F465457415245- <1>
3496 0001400E 20494E544552525550- <1>
3496 00014017 5400                <1>
3497                               <1> ifc_msg:
3498 00014019 0D0A                <1>             db 0Dh, 0Ah
3499 0001401B 494E56414C49442046- <1>             db "INVALID FUNCTION CALL"
3499 00014024 554E4354494F4E2043- <1>
3499 0001402D 414C4C              <1>
3500                               <1> inv_msg_for_trdos_v2:
3501 00014030 20                  <1>             db 20h
3502 00014031 666F72205452444F53- <1>             db "for TRDOS v2!"
3502 0001403A 20763221            <1>
```

```
3503 0001403E 07                  <1>             db 07h
3504 0001403F 0D0A                <1>             db 0Dh, 0Ah
3505 00014041 0D0A                <1>             db 0Dh, 0Ah
3506 00014043 494E5420            <1>             db "INT "
3507 00014047 303068              <1> int_num_str: db "00h"
3508 0001404A 0D0A                <1>             db 0Dh, 0Ah
3509 0001404C 454158203A20        <1>             db "EAX : "
3510 00014052 3030303030303030-   <1> eax_str:    db "00000000h", 0Dh, 0Ah
3510 0001405B 0D0A                <1>
3511 0001405D 454950203A20        <1>             db "EIP : "
3512 00014063 3030303030303030-   <1> eip_str:    db "00000000h", 0Dh, 0Ah, 0
3512 0001406C 0D0A00              <1>
3513                              <1>
3514                              <1> ; 17/04/2021
3515                              <1> ; ('KDEV' device parameters are disabled as temporary)
3516                              <1>
3517                              <1> ; 07/10/2016
3518                              <1> ; Device names & parameters (for kernel devices)
3519                              <1>
3520 0001406F 90                  <1> align 2
3521                              <1> ;KDEV_NAME:
3522                              <1> ;           db 'TTY',0,0,0,0,0 ; 1
3523                              <1> ;           db 'MEM',0,0,0,0,0 ; 2
3524                              <1> ;           db 'FD0',0,0,0,0,0 ; 3
3525                              <1> ;           db 'FD1',0,0,0,0,0 ; 4
3526                              <1> ;           db 'HD0',0,0,0,0,0 ; 5
3527                              <1> ;           db 'HD1',0,0,0,0,0 ; 6
3528                              <1> ;           db 'HD2',0,0,0,0,0 ; 7
3529                              <1> ;           db 'HD3',0,0,0,0,0 ; 8
3530                              <1> ;           db 'LPT',0,0,0,0,0 ; 9
3531                              <1> ;           db 'TTY0',0,0,0,0 ; 10
3532                              <1> ;           db 'TTY1',0,0,0,0 ; 11
3533                              <1> ;           db 'TTY2',0,0,0,0 ; 12
3534                              <1> ;           db 'TTY3',0,0,0,0 ; 13
3535                              <1> ;           db 'TTY4',0,0,0,0 ; 14
3536                              <1> ;           db 'TTY5',0,0,0,0 ; 15
3537                              <1> ;           db 'TTY6',0,0,0,0 ; 16
3538                              <1> ;           db 'TTY7',0,0,0,0 ; 17
3539                              <1> ;           db 'TTY8',0,0,0,0 ; 18
3540                              <1> ;           db 'TTY9',0,0,0,0 ; 19
3541                              <1> ;           db 'COM1',0,0,0,0 ; 18
3542                              <1> ;           db 'COM2',0,0,0,0 ; 19
3543                              <1> ;           ;db 'CONSOLE',0  ; 1
3544                              <1> ;           ;db 'PRINTER',0  ; 9
3545                              <1> ;           ;db 'CDROM'    ; 20
3546                              <1> ;           ;db 'CDROM0'   ; 20
3547                              <1> ;           ;db 'CDROM1'   ; 21
3548                              <1> ;           ;db 'DVD'      ; 22
3549                              <1> ;           ;db 'DVD0'     ; 22
3550                              <1> ;           ;db 'DVD1'     ; 23
3551                              <1> ;           ;db 'USB'      ; 24
3552                              <1> ;           ;db 'USB0'     ; 24
3553                              <1> ;           ;db 'USB1'     ; 25
3554                              <1> ;           ;db 'USB2'     ; 26
3555                              <1> ;           ;db 'USB3'     ; 27
3556                              <1> ;           ;db 'KEYBOARD' ; 1
3557                              <1> ;           ;db 'MOUSE'    ; 28
3558                              <1> ;           ;db 'SOUND'    ; 29
3559                              <1> ;           ;db 'VGA',0,0,0,0 ; 30
3560                              <1> ;           ;db 'CGA',0,0,0,0 ; 31
3561                              <1> ;           ;db 'AUDIO',0,0,0 ; 29
3562                              <1> ;           ;db 'VIDEO',0,0,0 ; 32
3563                              <1> ;           ;db 'MUSIC',0,0,0 ; 33
3564                              <1> ;           ;db 'ETHERNET'    ; 34
3565                              <1> ;           ;db 'SD0',0,0,0,0,0 ; 35
3566                              <1> ;           ;db 'SD1',0,0,0,0,0 ; 36
3567                              <1> ;           ;db 'SD2',0,0,0,0,0 ; 37
3568                              <1> ;           ;db 'SD3',0,0,0,0,0 ; 38
3569                              <1> ;           ;db 'SATA0'    ; 35
3570                              <1> ;           ;db 'SATA1'    ; 36
3571                              <1> ;           ;db 'SATA2'    ; 37
3572                              <1> ;           ;db 'SATA3'    ; 38
3573                              <1> ;           ;db 'PATA0',0,0,0  ; 5
3574                              <1> ;           ;db 'PATA1',0,0,0  ; 6
3575                              <1> ;           ;db 'PATA2',0,0,0  ; 7
3576                              <1> ;           ;db 'PATA3',0,0,0  ; 8
3577                              <1> ;           ;db 'WIRELESS'  ; 39
3578                              <1> ;           ;db 'HDMI',0,0,0,0 ; 40
3579                              <1> ;           db 'NULL',0,0,0,0 ; 0
3580                              <1>
3581                              <1> ;NumOfKernelDevNames equ ($-KDEV_NAME) / 8 ; 20 (07/10/2016)
3582                              <1>
3583                              <1> ;KDEV_NUMBER:
3584                              <1> ;           db 1,2,3,4,5,6,7,8,9
3585                              <1> ;           db 10,11,12,13,14,15,16,17,18,19
3586                              <1> ;           db 18,19,0
3587                              <1>
3588                              <1> ;NumOfKernelDevices equ $ - KDEV_NUMBER
3589                              <1>
3590                              <1> ;KDEV_OADDR:
3591                              <1> ;           dd otty ;tty  ; 1
3592                              <1> ;           dd sret ;mem  ; 2
3593                              <1> ;           dd sret ;fd0  ; 3
3594                              <1> ;           dd sret ;fd1  ; 4
3595                              <1> ;           dd sret ;hd0  ; 5
3596                              <1> ;           dd sret ;hd1  ; 6
3597                              <1> ;           dd sret ;hd2  ; 7
3598                              <1> ;           dd sret ;hd3  ; 8
3599                              <1> ;           dd sret ;lpt  ; 9
3600                              <1> ;           dd ocvt ;tty0 ; 10
3601                              <1> ;           dd ocvt ;tty1 ; 11
3602                              <1> ;           dd ocvt ;tty2 ; 12
3603                              <1> ;           dd ocvt ;tty3 ; 13
3604                              <1> ;           dd ocvt ;tty4 ; 14
3605                              <1> ;           dd ocvt ;tty5 ; 15
```

```
3606                                    <1> ;           dd ocvt ;tty6 ; 16
3607                                    <1> ;           dd ocvt ;tty7 ; 17
3608                                    <1> ;           dd ocvt ;tty8 ; 18
3609                                    <1> ;           dd ocvt ;tty9 ; 19
3610                                    <1> ;           ;dd ocvt ;com1 ; 18
3611                                    <1> ;           ;dd ocvt ;com2 ; 19
3612                                    <1> ;           dd sret ;null ; 20
3613                                    <1> ;KDEV_CADDR:
3614                                    <1> ;           dd ctty ;tty  ; 1
3615                                    <1> ;           dd cret ;mem  ; 2
3616                                    <1> ;           dd cret ;fd0  ; 3
3617                                    <1> ;           dd cret ;fd1  ; 4
3618                                    <1> ;           dd cret ;hd0  ; 5
3619                                    <1> ;           dd cret ;hd1  ; 6
3620                                    <1> ;           dd cret ;hd2  ; 7
3621                                    <1> ;           dd cret ;hd3  ; 8
3622                                    <1> ;           dd cret ;lpt  ; 9
3623                                    <1> ;           dd ocvt ;tty0 ; 10
3624                                    <1> ;           dd ccvt ;tty1 ; 11
3625                                    <1> ;           dd ccvt ;tty2 ; 12
3626                                    <1> ;           dd ccvt ;tty3 ; 13
3627                                    <1> ;           dd ccvt ;tty4 ; 14
3628                                    <1> ;           dd ccvt ;tty5 ; 15
3629                                    <1> ;           dd ccvt ;tty6 ; 16
3630                                    <1> ;           dd ccvt ;tty7 ; 17
3631                                    <1> ;           dd ccvt ;tty8 ; 18
3632                                    <1> ;           dd ccvt ;tty9 ; 19
3633                                    <1> ;           ;dd ccvt ;com1 ; 18
3634                                    <1> ;           ;dd ccvt ;com2 ; 19
3635                                    <1> ;           dd cret ;null ; 20
3636                                    <1> ;
3637                                    <1> ;KDEV_RADDR:
3638                                    <1> ;           dd rtty ;tty  ; 1
3639                                    <1> ;           dd rmem ;mem  ; 2
3640                                    <1> ;           dd rfd  ;fd0  ; 3
3641                                    <1> ;           dd rfd  ;fd1  ; 4
3642                                    <1> ;           dd rhd  ;hd0  ; 5
3643                                    <1> ;           dd rhd  ;hd1  ; 6
3644                                    <1> ;           dd rhd  ;hd2  ; 7
3645                                    <1> ;           dd rhd  ;hd3  ; 8
3646                                    <1> ;           dd rlpt ;lpt  ; 9
3647                                    <1> ;           dd rcvt ;tty0 ; 10
3648                                    <1> ;           dd rcvt ;tty1 ; 11
3649                                    <1> ;           dd rcvt ;tty2 ; 12
3650                                    <1> ;           dd rcvt ;tty3 ; 13
3651                                    <1> ;           dd rcvt ;tty4 ; 14
3652                                    <1> ;           dd rcvt ;tty5 ; 15
3653                                    <1> ;           dd rcvt ;tty6 ; 16
3654                                    <1> ;           dd rcvt ;tty7 ; 17
3655                                    <1> ;           dd rcvt ;tty8 ; 18
3656                                    <1> ;           dd rcvt ;tty9 ; 19
3657                                    <1> ;           ;dd rcvt ;com1 ; 18
3658                                    <1> ;           ;dd rcvt ;com2 ; 19
3659                                    <1> ;           dd rnull ;null ; 20
3660                                    <1> ;KDEV_WADDR:
3661                                    <1> ;           dd wtty ;tty  ; 1
3662                                    <1> ;           dd wmem ;mem  ; 2
3663                                    <1> ;           dd wfd  ;fd0  ; 3
3664                                    <1> ;           dd wfd  ;fd1  ; 4
3665                                    <1> ;           dd whd  ;hd0  ; 5
3666                                    <1> ;           dd whd  ;hd1  ; 6
3667                                    <1> ;           dd whd  ;hd2  ; 7
3668                                    <1> ;           dd whd  ;hd3  ; 8
3669                                    <1> ;           dd wlpt ;lpt  ; 9
3670                                    <1> ;           dd xmtt ;tty0 ; 10
3671                                    <1> ;           dd xmtt ;tty1 ; 11
3672                                    <1> ;           dd xmtt ;tty2 ; 12
3673                                    <1> ;           dd xmtt ;tty3 ; 13
3674                                    <1> ;           dd xmtt ;tty4 ; 14
3675                                    <1> ;           dd xmtt ;tty5 ; 15
3676                                    <1> ;           dd xmtt ;tty6 ; 16
3677                                    <1> ;           dd xmtt ;tty7 ; 17
3678                                    <1> ;           dd xmtt ;tty8 ; 18
3679                                    <1> ;           dd xmtt ;tty9 ; 19
3680                                    <1> ;           ;dd xmtt ;com1 ; 18
3681                                    <1> ;           ;dd xmtt ;com2 ; 19
3682                                    <1> ;           dd wnull ;null ; 20
3683                                    <1>
3684                                    <1> ; DEV_ACCESS bits:
3685                                    <1>       ; bit 0 = accessable by normal users
3686                                    <1>       ; bit 1 = read access permission
3687                                    <1>       ; bit 2 = write access permission
3688                                    <1>       ; bit 3 = IOCTL permission to users
3689                                    <1>       ; bit 4 = block device if it is set
3690                                    <1>       ; bit 5 = 16 bit or 1024 byte data
3691                                    <1>       ; bit 6 = 32 bit or 2048 byte data
3692                                    <1>       ; bit 7 = installable device driver
3693                                    <1>
3694                                    <1> ;KDEV_ACCESS: ; 08/10/2016
3695                                    <1> ;           db  00000111b; tty, 1
3696                                    <1> ;           db  00000111b; mem, 2
3697                                    <1> ;           db  10001111b; fd0, 3
3698                                    <1> ;           db  10001111b; fd1, 4
3699                                    <1> ;           db  10001111b; hd0, 5
3700                                    <1> ;           db  10001111b; hd1, 6
3701                                    <1> ;           db  10001111b; hd2, 7
3702                                    <1> ;           db  10001111b; hd3, 8
3703                                    <1> ;           db  00000111b   ; lpt, 9
3704                                    <1> ;           db  00000111b; tty0, 10
3705                                    <1> ;           db  00000111b; tty1, 11
3706                                    <1> ;           db  00000111b; tty2, 12
3707                                    <1> ;           db  00000111b; tty3, 13
3708                                    <1> ;           db  00000111b; tty4, 14
3709                                    <1> ;           db  00000111b; tty5, 15
3710                                    <1> ;           db  00000111b; tty6, 16
```

```
3711                                 <1> ;           db  00000111b; tty7, 17
3712                                 <1> ;           db  00000111b; tty8, 18
3713                                 <1> ;           db  00000111b; tty9, 19
3714                                 <1> ;           ;db 00000111b; com1, 18
3715                                 <1> ;           ;db 00000111b; com2, 19
3716                                 <1> ;           db  00000000b   ; null, 0
3717                                 <1>
3718                                 <1> ; 07/10/2016
3719                                 <1> ;NumOfInstallableDevices equ 8
3720                                 <1> ;NUMIDEV    equ NumOfInstallableDevices ; 8
3721                                 <1> ;NUMOFDEVICES      equ NumOfKernelDevices + NumOfInstallableDevices
3722                                 <1>
3723                                 <1> ; 26/02/2017
3724                                 <1> ; IRQ Callback (& Signal Response Byte) service availibity
3725                                 <1> ; 'syscalbac'
3726                                 <1> ; ************************************************
3727                                 <1> ; IRQ 00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
3728                                 <1> ; --- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
3729                                 <1> ; --- 00 00 00 01 02 03 00 04 00 05 06 07 08 09 00 00
3730                                 <1> ; ************************************************
3731                                 <1> IRQenum:
3732 00014070 000000010203000400-    <1>       db  0,0,0,1,2,3,0,4,0,5,6,7,8,9,0,0
3732 00014079 05060708090000         <1>
3733                                 <1>
3734                                 <1> ; 28/08/2017
3735                                 <1> ; 20/08/2017
3736                                 <1> ; DMA Registers (for 'sysdma')
3737                                 <1> ; 02/07/2017 (sb16mod.s)
3738 00014080 00020406C0C4C8CC       <1> dma_adr:    db 0,2,4,6,0C0h,0C4h,0C8h,0CCh
3739 00014088 01030507C2C6CACE       <1> dma_cnt:    db 1,3,5,7,0C2h,0C6h,0CAh,0CEh
3740 00014090 878381828F8B898A       <1> dma_page:   db 87h,83h,81h,82h,8Fh,8Bh,89h,8Ah ; 03/08/2017
3741 00014098 0A0A0A0AD4D4D4D4       <1> dma_mask:   db 0Ah,0Ah,0Ah,0Ah,0D4h,0D4h,0D4h,0D4h
3742 000140A0 0B0B0B0BD6D6D6D6       <1> dma_mod:    db 0Bh,0Bh,0Bh,0Bh,0D6h,0D6h,0D6h,0D6h
3743 000140A8 0C0C0C0CD8D8D8D8       <1> dma_flip:   db 0Ch,0Ch,0Ch,0Ch,0D8h,0D8h,0D8h,0D8h
3115
3116                                     ; 27/08/2014
3117                                     scr_row:
3118 000140B0 E0810B00                       dd 0B8000h + 0A0h + 0A0h + 0A0h ; Row 3
3119                                     scr_col:
3120 000140B4 00000000                       dd 0
3121
3122                                     Align 4
3123                                         ; 15/04/2016
3124                                         ; TRDOS 386 (TRDOS v2.0)
3125
3126                                         ; 21/08/2014
3127                                     ilist:
3128                                         ;times       32 dd cpu_except ; INT 0 to INT 1Fh
3129                                         ;
3130                                         ; Exception list
3131                                         ; 25/08/2014
3132 000140B8 [DC0B0000]                    dd     exc0   ; 0h,  Divide-by-zero Error
3133 000140BC [E30B0000]                    dd     exc1
3134 000140C0 [EA0B0000]                    dd     exc2
3135 000140C4 [F10B0000]                    dd     exc3
3136 000140C8 [F50B0000]                    dd     exc4
3137 000140CC [F90B0000]                    dd     exc5
3138 000140D0 [FD0B0000]                    dd     exc6   ; 06h,  Invalid Opcode
3139 000140D4 [010C0000]                    dd     exc7
3140 000140D8 [050C0000]                    dd     exc8
3141 000140DC [090C0000]                    dd     exc9
3142 000140E0 [0D0C0000]                    dd     exc10
3143 000140E4 [110C0000]                    dd     exc11
3144 000140E8 [150C0000]                    dd     exc12
3145 000140EC [190C0000]                    dd     exc13  ; 0Dh, General Protection Fault
3146 000140F0 [1D0C0000]                    dd     exc14  ; 0Eh, Page Fault
3147 000140F4 [210C0000]                    dd     exc15
3148 000140F8 [250C0000]                    dd     exc16
3149 000140FC [290C0000]                    dd     exc17
3150 00014100 [2D0C0000]                    dd     exc18
3151 00014104 [310C0000]                    dd     exc19
3152 00014108 [350C0000]                    dd     exc20
3153 0001410C [390C0000]                    dd     exc21
3154 00014110 [3D0C0000]                    dd     exc22
3155 00014114 [410C0000]                    dd     exc23
3156 00014118 [450C0000]                    dd     exc24
3157 0001411C [490C0000]                    dd     exc25
3158 00014120 [4D0C0000]                    dd     exc26
3159 00014124 [510C0000]                    dd     exc27
3160 00014128 [550C0000]                    dd     exc28
3161 0001412C [590C0000]                    dd     exc29
3162 00014130 [5D0C0000]                    dd     exc30
3163 00014134 [610C0000]                    dd     exc31
3164                                    IRQ_list: ; 28/02/2017 ('syscalbac')
3165                                         ; Interrupt list
3166 00014138 [50090000]                    dd     timer_int    ; INT 20h
3167                                             ;dd    irq0
3168 0001413C [BE100000]                    dd     kb_int       ; 24/01/2016
3169                                             ;dd    irq1
3170 00014140 [320B0000]                    dd     irq2
3171                                             ; COM2 int
3172 00014144 [360B0000]                    dd     irq3
3173                                             ; COM1 int
3174 00014148 [410B0000]                    dd     irq4
3175 0001414C [4C0B0000]                    dd     irq5
3176                                    ;DISKETTE_INT: ;06/02/2015
3177 00014150 [CE500000]                    dd     fdc_int            ; 16/02/2015, IRQ 6 handler
3178                                             ;dd    irq6
3179                                    ; Default IRQ 7 handler against spurious IRQs (from master PIC)
3180                                    ; 25/02/2015 (source: http://wiki.osdev.org/8259_PIC)
3181 00014154 [B90E0000]                    dd     default_irq7 ; 25/02/2015
3182                                             ;dd    irq7
3183                                    ; Real Time Clock Interrupt
3184 00014158 [BB0A0000]                    dd     rtc_int            ; 23/02/2015, IRQ 8 handler
3185                                             ;dd    irq8   ; INT 28h
```

```
3186 0001415C [5C0B0000]                    dd     irq9
3187 00014160 [600B0000]                    dd     irq10
3188 00014164 [640B0000]                    dd     irq11
3189 00014168 [680B0000]                    dd     irq12
3190 0001416C [6C0B0000]                    dd     irq13
3191                                  ;HDISK_INT1:  ;06/02/2015
3192 00014170 [2E5A0000]                    dd     hdc1_int      ; 21/02/2015, IRQ 14 handler
3193                                        ;dd    irq14
3194                                  ;HDISK_INT2:  ;06/02/2015
3195 00014174 [515A0000]                    dd     hdc2_int      ; 21/02/2015, IRQ 15 handler
3196                                        ;dd    irq15  ; INT 2Fh
3197                                        ; 14/08/2015
3198                                        ;dd    sysent        ; INT 30h (system calls)
3199
3200                                        ; 15/04/2016
3201                                        ; TRDOS 386(TRDOS v2.0) Software Interrupts
3202
3203 00014178 [D5410100]                    dd     int30h        ; Reserved for
3204                                                              ; !!! Retro UNIX (RUNIX) !!!
3205                                                              ; !!! SINGLIX !!! System Calls
3206 0001417C [53170000]                    dd     int31h        ; Video BIOS (IBM PC/AT, Int 10h)
3207 00014180 [E30E0000]                    dd     int32h        ; Keyboard Functions (IBM PC/AT, Int 16h)
3208 00014184 [79510000]                    dd     int33h        ; DISK I/O (IBM PC/AT, Int 13h)
3209 00014188 [36270100]                    dd     int34h        ; #IOCTL# (I/O port access support for ring 3)
3210 0001418C [3B650000]                    dd     int35h        ; Time/Date Functions (IBM PC/AT, Int 1Ah)
3211 00014190 [6D0D0000]                    dd     ignore_int    ; INT 36h : Timer Functions
3212 00014194 [6D0D0000]                    dd     ignore_int    ; INT 37h
3213 00014198 [6D0D0000]                    dd     ignore_int    ; INT 38h
3214 0001419C [6D0D0000]                    dd     ignore_int    ; INT 39h
3215 000141A0 [6D0D0000]                    dd     ignore_int    ; INT 3Ah
3216 000141A4 [6D0D0000]                    dd     ignore_int    ; INT 3Bh
3217 000141A8 [6D0D0000]                    dd     ignore_int    ; INT 3Ch
3218 000141AC [6D0D0000]                    dd     ignore_int    ; INT 3Dh
3219 000141B0 [6D0D0000]                    dd     ignore_int    ; INT 3Eh
3220 000141B4 [6D0D0000]                    dd     ignore_int    ; INT 3Fh
3221 000141B8 [7ED20000]                    dd     sysent        ; INT 40h : !!! TRDOS 386 System Calls !!!
3222                                        ;dd    ignore_int
3223 000141BC 00000000                      dd     0
3224
3225                                  ; 20/08/2014
3226                                    ; /* This is the default interrupt "handler" :-) */
3227                                    ; Linux v0.12 (head.s)
3228                                  int_msg:
3229 000141C0 556E6B6E6F776E2069-           db "Unknown interrupt ! ", 0
3229 000141C9 6E7465727275707420-
3229 000141D2 212000
3230
3231                                  ; 15/04/2016
3232                                  ; TRDOS 386 (TRDOS v2.0)
3233
3234                                  ; 29/04/2016
3235                                  int30h:
3236                                  trdos_isc_routine:
3237                                        ; 02/05/2016
3238                                        ; 01/05/2016
3239                                        ; 29/04/2016
3240                                        ; 18/04/2016
3241                                        ; 15/04/2016 (TRDOS 386 = TRDOS v2.0)
3242                                        ; 17/04/2011 (TRDOS v1.0, 'IFC.ASM')
3243                                        ; 03/02/2011 ('trdos_ifc_routine')
3244                                        ;
3245 000141D5 8B1C24                      mov    ebx, [esp] ; EIP (next)
3246 000141D8 83EB02                      sub    ebx, 2 ; EIP (CD ##h)
3247
3248 000141DB 89C1                        mov    ecx, eax
3249 000141DD 8A4301                      mov    al, [ebx+1] ; CDh ##h
3250
3251 000141E0 66BA1000                    mov    dx, KDATA
3252 000141E4 8EDA                        mov    ds, dx
3253 000141E6 8EC2                        mov    es, dx
3254
3255 000141E8 FC                          cld
3256 000141E9 8B15[A8800100]              mov    edx, [k_page_dir]
3257 000141EF 0F22DA                      mov    cr3, edx
3258
3259 000141F2 E81800FFFF                  call   bytetohex
3260 000141F7 66A3[47400100]              mov    [int_num_str], ax
3261
3262 000141FD 89D8                        mov    eax, ebx ; EIP
3263 000141FF E84B00FFFF                  call   dwordtohex
3264 00014204 8915[63400100]              mov    [eip_str], edx
3265 0001420A A3[67400100]                mov    [eip_str+4], eax
3266
3267 0001420F 89C8                        mov    eax, ecx
3268 00014211 E83900FFFF                  call   dwordtohex
3269 00014216 8915[52400100]              mov    [eax_str], edx
3270 0001421C A3[56400100]                mov    [eax_str+4], eax
3271
3272 00014221 43                          inc    ebx
3273 00014222 8A03                        mov    al, [ebx] ; Interrupt number
3274
3275                                  trdos_isc_handler:
3276 00014224 80FE30                      cmp    dh, 30h ; Retro UNIX, SINGLIX System calls
3277 00014227 7507                        jne    short trdos_usi_handler
3278 00014229 BE[E43F0100]                mov    esi, isc_msg
3279 0001422E EB05                        jmp    short loc_write_inv_system_call_msg
3280
3281                                  trdos_usi_handler:
3282 00014230 BE[FA3F0100]                mov    esi, usi_msg
3283
3284                                  loc_write_inv_system_call_msg:
3285 00014235 E80B2EFFFF                  call   print_msg
3286                                      ; 29/04/2016
3287 0001423A BE[30400100]                mov    esi, inv_msg_for_trdos_v2
3288 0001423F E8012EFFFF                  call   print_msg
```

```
                                loc_ifc_terminate_process:
                                      ; u.uno = process number
                                      ; 29/04/2016
                                
                                      ; 02/05/2016
00014244 FE05[5B030300]               inc   byte [sysflg] ; 0FFh -> 0
                                
0001424A B801000000                   mov   eax, 1
0001424F E90393FFFF                   jmp   sysexit
                                
                                ; 07/03/2015
                                ; Temporary Code
                                display_disks:
00014254 803D[EE680000]00             cmp   byte [fd0_type], 0
0001425B 7605                         jna   short ddsks1
0001425D E87D000000                   call  pdskm
                                ddsks1:
00014262 803D[EF680000]00             cmp   byte [fd1_type], 0
00014269 760C                         jna   short ddsks2
0001426B C605[AF430100]31             mov   byte [dskx], '1'
00014272 E868000000                   call  pdskm
                                ddsks2:
00014277 803D[F0680000]00             cmp   byte [hd0_type], 0
0001427E 7654                         jna   short ddsk6
00014280 66C705[AD430100]68-          mov   word [dsktype], 'hd'
00014288 64
00014289 C605[AF430100]30             mov   byte [dskx], '0'
00014290 E84A000000                   call  pdskm
                                ddsks3:
00014295 803D[F1680000]00             cmp   byte [hd1_type], 0
0001429C 7636                         jna   short ddsk6
0001429E C605[AF430100]31             mov   byte [dskx], '1'
000142A5 E835000000                   call  pdskm
                                ddsks4:
000142AA 803D[F2680000]00             cmp   byte [hd2_type], 0
000142B1 7621                         jna   short ddsk6
000142B3 C605[AF430100]32             mov   byte [dskx], '2'
000142BA E820000000                   call  pdskm
                                ddsks5:
000142BF 803D[F3680000]00             cmp   byte [hd3_type], 0
000142C6 760C                         jna   short ddsk6
000142C8 C605[AF430100]33             mov   byte [dskx], '3'
000142CF E80B000000                   call  pdskm
                                ddsk6:
000142D4 BE[D7430100]                 mov   esi, nextline
000142D9 E806000000                   call  pdskml
                                pdskm_ok:
000142DE C3                           retn
                                pdskm:
000142DF BE[AB430100]                 mov   esi, dsk_ready_msg
                                pdskml:
000142E4 AC                           lodsb
000142E5 08C0                         or    al, al
000142E7 74F5                         jz    short pdskm_ok
000142E9 56                           push  esi
                                      ; 13/05/2016
000142EA BB07000000                   mov   ebx, 7  ; Black background,
                                                    ; light gray forecolor
                                                    ; Video page 0 (bh=0)
000142EF E86FDFFEFF                   call  _write_tty
000142F4 5E                           pop   esi
000142F5 EBED                         jmp   short pdskml
                                
000142F7 90                     Align 2
                                      ; 21/08/2014
                                exc_msg:
000142F8 435055206578636570-          db "CPU exception ! "
00014301 74696F6E202120
                                excnstr:          ; 25/08/2014
00014308 3F3F68202045495020-          db "??h", "  EIP : "
00014311 3A20
                                EIPstr: ; 29/08/2014
00014313 00<rep Ch>                   times 12 db 0
                                
                                      ; 23/02/2015
                                      ; 25/08/2014
                                ;scounter:
                                ;     db 5
                                ;     db 19
                                
                                ; 06/11/2014
                                ; Memory Information message
                                ; 14/08/2015
                                msg_memory_info:
0001431F 07                           db    07h
00014320 0D0A                         db    0Dh, 0Ah
                                      ;db   "MEMORY ALLOCATION INFO", 0Dh, 0Ah, 0Dh, 0Ah
00014322 546F74616C206D656D-          db    "Total memory : "
0001432B 6F7279203A20
                                mem_total_b_str: ; 10 digits
00014331 303030303030303030-          db    "0000000000 bytes", 0Dh, 0Ah
0001433A 302062797465730D0A
00014343 202020202020202020-          db    "               ", 20h, 20h, 20h
0001434C 202020202020202020
                                mem_total_p_str: ; 7 digits
00014355 303030303030302070-          db    "0000000 pages", 0Dh, 0Ah
0001435E 616765730D0A
00014364 0D0A                         db    0Dh, 0Ah
00014366 46726565206D656D6F-          db    "Free memory  : "
0001436F 727920203A20
                                free_mem_b_str:  ; 10 digits
00014375 3F3F3F3F3F3F3F3F3F-          db    "?????????? bytes", 0Dh, 0Ah
0001437E 3F2062797465730D0A
00014387 202020202020202020-          db    "               ", 20h, 20h, 20h
```

```
3384 00014390 202020202020202020
3385                                free_mem_p_str:  ; 7 digits
3386 00014399 3F3F3F3F3F3F3F2070-           db     "??????? pages", 0Dh, 0Ah
3386 000143A2 616765730D0A
3387 000143A8 0D0A00                        db     0Dh, 0Ah, 0
3388
3389                                dsk_ready_msg:
3390 000143AB 0D0A                          db     0Dh, 0Ah
3391                                dsktype:
3392 000143AD 6664                          db     'fd'
3393                                dskx:
3394 000143AF 30                            db     '0'
3395 000143B0 20                            db     20h
3396 000143B1 697320524541445920-           db     'is READY ...'
3396 000143BA 2E2E2E
3397 000143BD 00                            db     0
3398
3399                                setup_error_msg:
3400 000143BE 0D0A                          db     0Dh, 0Ah
3401 000143C0 4469736B2053657475-           db     'Disk Setup Error !'
3401 000143C9 70204572726F722021
3402 000143D2 0D0A00                        db     0Dh, 0Ah,0
3403
3404                                next2line: ; 08/02/2016
3405 000143D5 0D0A                          db     0Dh, 0Ah
3406                                nextline:
3407 000143D7 0D0A00                        db     0Dh, 0Ah, 0
3408
3409                                ; temporary
3410                                ; 19/12/2020
3411                                msg_lfb_addr:
3412                                        ;db    0Dh, 0Ah
3413 000143DA 4C696E656172206672-           db     "Linear frame buffer at "
3413 000143E3 616D65206275666665-
3413 000143EC 7220617420
3414                                lfb_addr_str: ; 8 (hex) digits
3415 000143F1 303030303030303068-           db     "00000000h", 0Dh, 0Ah
3415 000143FA 0D0A
3416 000143FC 0D0A00                        db     0Dh, 0Ah, 0
3417
3418                                ; KERNEL - SYSINIT Messages
3419                                ; 24/08/2015
3420                                ; 13/04/2015 - (Retro UNIX 386 v1 Beginning)
3421                                ; 14/07/2013
3422                                ;kernel_init_err_msg:
3423                                ;     db 0Dh, 0Ah
3424                                ;     db 07h
3425                                ;     db 'Kernel initialization ERROR !'
3426                                ;     db 0Dh, 0Ah, 0
3427
3428                                ;welcome_msg:
3429                                ;     db 0Dh, 0Ah
3430                                ;     db 07h
3431                                ;     db 'Welcome to TRDOS 386 Operating System !'
3432                                ;     db 0Dh, 0Ah
3433                                ;     db 'by Erdogan Tan - 31/12/2017 (v2.0.0)'
3434                                ;     db 0Dh, 0Ah, 0
3435
3436                                panic_msg:
3437 000143FF 0D0A07                        db 0Dh, 0Ah, 07h
3438 00014402 4552524F523A204B65-           db 'ERROR: Kernel Panic !'
3438 0001440B 726E656C2050616E69-
3438 00014414 632021
3439 00014417 0D0A00                        db 0Dh, 0Ah, 0
3440
3441                                ;msgl_drv_not_ready:
3442                                ;     db 07h, 0Dh, 0Ah
3443                                ;       db 'Drive not ready or read error !'
3444                                ;       db 0Dh, 0Ah, 0
3445
3446                                starting_msg:
3447 0001441A 5475726B6973682052-           db "Turkish Rational DOS v2.0 [18/04/2021] ...", 0
3447 00014423 6174696F6E616C2044-
3447 0001442C 4F532076322E30205B-
3447 00014435 31382F30342F323032-
3447 0001443E 315D202E2E2E00
3448                                    NextLine:
3449 00014445 0D0A00                        db 0Dh, 0Ah, 0
3450
3451                                    %include 'audio.s' ; 03/04/2017
3452                                <1> ; ****************************************************************************
3453                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.2 - audio.s
3454                                <1> ; ----------------------------------------------------------------------------
3455                                <1> ; Last Update: 01/09/2020
3456                                <1> ; ----------------------------------------------------------------------------
3457                                <1> ; Beginning: 03/04/2017
3458                                <1> ; ----------------------------------------------------------------------------
3459                                <1> ; Assembler: NASM version 2.11 (trdos386.s)
3460                                <1> ; ****************************************************************************
3461                                <1>
3462                                <1> ; AUDIO CONTROLLER & CODEC DEFINITIONS & CODE FOR TRDOS 386
3463                                <1>
3464                                <1> ;============================================================================
3465                                <1> ;               EQUATES
3466                                <1> ;============================================================================
3467                                <1>
3468                                <1> ; PCI EQUATES
3469                                <1>
3470                                <1> BIT0  EQU 1
3471                                <1> BIT1  EQU 2
3472                                <1> BIT2  EQU 4
3473                                <1> BIT3  EQU 8
3474                                <1> BIT4  EQU 10h
3475                                <1> BIT5  EQU 20h
3476                                <1> BIT6  EQU 40h
```

```
3477                                    <1> BIT7  EQU 80h
3478                                    <1> BIT8  EQU 100h
3479                                    <1> BIT9  EQU 200h
3480                                    <1> BIT10 EQU 400h
3481                                    <1> BIT11 EQU 800h
3482                                    <1> BIT12 EQU 1000h
3483                                    <1> BIT13 EQU 2000h
3484                                    <1> BIT14 EQU 4000h
3485                                    <1> BIT15 EQU 8000h
3486                                    <1> BIT16 EQU 10000h
3487                                    <1> BIT17 EQU 20000h
3488                                    <1> BIT18 EQU 40000h
3489                                    <1> BIT19 EQU 80000h
3490                                    <1> BIT20 EQU 100000h
3491                                    <1> BIT21 EQU 200000h
3492                                    <1> BIT22 EQU 400000h
3493                                    <1> BIT23 EQU 800000h
3494                                    <1> BIT24 EQU 1000000h
3495                                    <1> BIT25 EQU 2000000h
3496                                    <1> BIT26 EQU 4000000h
3497                                    <1> BIT27 EQU 8000000h
3498                                    <1> BIT28 EQU 10000000h
3499                                    <1> BIT29 EQU 20000000h
3500                                    <1> BIT30 EQU 40000000h
3501                                    <1> BIT31 EQU 80000000h
3502                                    <1> NOT_BIT31 EQU 7FFFFFFFh
3503                                    <1>
3504                                    <1> ; PCI equates
3505                                    <1> ; PCI function address (PFA)
3506                                    <1> ; bit 31 = 1
3507                                    <1> ; bit 23:16 = bus number     (0-255)
3508                                    <1> ; bit 15:11 = device number  (0-31)
3509                                    <1> ; bit 10:8 = function number (0-7)
3510                                    <1> ; bit 7:0 = register number  (0-255)
3511                                    <1>
3512                                    <1> IO_ADDR_MASK    EQU     0FFFEh  ; mask off bit 0 for reading BARs
3513                                    <1> PCI_INDEX_PORT  EQU     0CF8h
3514                                    <1> PCI_DATA_PORT   EQU     0CFCh
3515                                    <1> PCI32           EQU     BIT31   ; bitflag to signal 32bit access
3516                                    <1> PCI16           EQU     BIT30   ; bitflag for 16bit access
3517                                    <1> NOT_PCI32_PCI16 EQU     03FFFFFFh ; NOT BIT31+BIT30 ; 19/03/2017
3518                                    <1>
3519                                    <1> PCI_FN0         EQU     0 << 8
3520                                    <1> PCI_FN1         EQU     1 << 8
3521                                    <1> PCI_FN2         EQU     2 << 8
3522                                    <1> PCI_FN3         EQU     3 << 8
3523                                    <1> PCI_FN4         EQU     4 << 8
3524                                    <1> PCI_FN5         EQU     5 << 8
3525                                    <1> PCI_FN6         EQU     6 << 8
3526                                    <1> PCI_FN7         EQU     7 << 8
3527                                    <1>
3528                                    <1> PCI_CMD_REG        EQU    04h    ; reg 04, command reg
3529                                    <1> IO_ENA             EQU    BIT0   ; i/o decode enable
3530                                    <1> MEM_ENA            EQU    BIT1   ; memory decode enable
3531                                    <1> BM_ENA             EQU    BIT2   ; bus master enable
3532                                    <1>
3533                                    <1> ; VIA VT8233 EQUATES
3534                                    <1>
3535                                    <1> VIA_VID            equ 1106h    ; VIA's PCI vendor ID
3536                                    <1> VT8233_DID         equ 3059h    ; VT8233 (VT8235) device ID
3537                                    <1>
3538                                    <1> PCI_IO_BASE          equ 10h
3539                                    <1> AC97_INT_LINE        equ 3Ch
3540                                    <1> VIA_ACLINK_CTRL      equ 41h
3541                                    <1> VIA_ACLINK_STAT      equ 40h
3542                                    <1> VIA_ACLINK_C00_READY equ 01h ; primary codec ready
3543                                    <1>
3544                                    <1> VIA_REG_AC97     equ 80h ; dword
3545                                    <1>
3546                                    <1> VIA_ACLINK_CTRL_ENABLE    equ   80h ; 0: disable, 1: enable
3547                                    <1> VIA_ACLINK_CTRL_RESET     equ   40h ; 0: assert, 1: de-assert
3548                                    <1> VIA_ACLINK_CTRL_SYNC      equ   20h ; 0: release SYNC, 1: force SYNC hi
3549                                    <1> VIA_ACLINK_CTRL_VRA       equ   08h ; 0: disable VRA, 1: enable VRA
3550                                    <1> VIA_ACLINK_CTRL_PCM       equ   04h ; 0: disable PCM, 1: enable PCM
3551                                    <1>                                     ; 3D Audio Channel slots 3/4
3552                                    <1> VIA_ACLINK_CTRL_INIT      equ  (VIA_ACLINK_CTRL_ENABLE +
VIA_ACLINK_CTRL_RESET +                                 VIA_ACLINK_CTRL_PCM +                                   VIA_ACLINK_CTRL_VRA)
3556                                    <1>
3557                                    <1> CODEC_AUX_VOL             equ   04h
3558                                    <1> VIA_REG_AC97_BUSY  equ   01000000h ;(1<<24)
3559                                    <1> VIA_REG_AC97_CMD_SHIFT    equ   10h ; 16
3560                                    <1> VIA_REG_AC97_PRIMARY_VALID equ 02000000h ;(1<<25)
3561                                    <1> VIA_REG_AC97_READ  equ   00800000h ;(1<<23)
3562                                    <1> VIA_REG_AC97_CODEC_ID_SHIFT   equ  1Eh ; 30
3563                                    <1> VIA_REG_AC97_CODEC_ID_PRIMARY equ  0
3564                                    <1> VIA_REG_AC97_DATA_SHIFT equ   0
3565                                    <1> VIADEV_PLAYBACK          equ   0
3566                                    <1> VIA_REG_OFFSET_STATUS   equ   0    ;; byte - channel status
3567                                    <1> VIA_REG_OFFSET_CONTROL  equ   01h  ;; byte - channel control
3568                                    <1> VIA_REG_CTRL_START      equ   80h  ;; WO
3569                                    <1> VIA_REG_CTRL_TERMINATE  equ   40h  ;; WO
3570                                    <1> VIA_REG_CTRL_PAUSE      equ   08h  ;; RW
3571                                    <1> VIA_REG_CTRL_RESET      equ   01h  ;; RW - probably reset? undocumented
3572                                    <1> VIA_REG_OFFSET_STOP_IDX equ   08h  ;; dword - stop index, channel type, sample rate
3573                                    <1> VIA8233_REG_TYPE_16BIT  equ   200000h ;; RW
3574                                    <1> VIA8233_REG_TYPE_STEREO equ   100000h ;; RW
3575                                    <1> VIA_REG_OFFSET_CURR_INDEX equ 0Fh ;; byte - channel current index (for via8233 only)
3576                                    <1> VIA_REG_OFFSET_TABLE_PTR equ  04h  ;; dword - channel table pointer
3577                                    <1> VIA_REG_OFFSET_CURR_PTR equ   04h  ;; dword - channel current pointer
3578                                    <1> VIA_REG_OFS_PLAYBACK_VOLUME_L equ  02h ;; byte
3579                                    <1> VIA_REG_OFS_PLAYBACK_VOLUME_R equ  03h ;; byte
3580                                    <1> VIA_REG_CTRL_AUTOSTART    equ   20h
3581                                    <1> VIA_REG_CTRL_INT_EOL      equ   02h
3582                                    <1> VIA_REG_CTRL_INT_FLAG     equ   01h
```

```
3583                                    <1> VIA_REG_CTRL_INT   equ  (VIA_REG_CTRL_INT_FLAG +                                VIA_REG_CTRL_INT_EOL
+                                   VIA_REG_CTRL_AUTOSTART)
3586                                    <1>
3587                                    <1> VIA_REG_STAT_STOP_IDX equ   10h    ;; RO ; 27/07/2020
3588                                    <1>                               ; current index = stop index
3589                                    <1> VIA_REG_STAT_STOPPED equ   04h    ;; RWC
3590                                    <1> VIA_REG_STAT_EOL   equ   02h    ;; RWC
3591                                    <1> VIA_REG_STAT_FLAG  equ   01h    ;; RWC
3592                                    <1> VIA_REG_STAT_ACTIVE equ   80h    ;; RO
3593                                    <1> ; 28/11/2016
3594                                    <1> VIA_REG_STAT_LAST  equ   40h    ;; RO
3595                                    <1> VIA_REG_STAT_TRIGGER_QUEUED equ 08h  ;; RO
3596                                    <1> VIA_REG_CTRL_INT_STOP     equ   04h  ; Interrupt on Current Index = Stop Index
3597                                    <1>                                  ; and End of Block
3598                                    <1>
3599                                    <1> VIA_REG_OFFSET_CURR_COUNT equ 0Ch ;; dword - channel current count, index
3600                                    <1>
3601                                    <1> PORTB              EQU    061h
3602                                    <1> REFRESH_STATUS     EQU    010h   ; Refresh signal status
3603                                    <1>
3604                                    <1> ; AC97 Codec registers.
3605                                    <1>
3606                                    <1> ; 22/07/2020
3607                                    <1> ; REALTEK ALC655 and ADI SOUNDMAX AD1980 CODEC MIXER REGISTERS
3608                                    <1>
3609                                    <1> ; each codec/mixer register is 16bits
3610                                    <1>
3611                                    <1> CODEC_RESET_REG                equ     00h  ; reset codec
3612                                    <1> CODEC_MASTER_VOL_REG           equ     02h  ; master volume
3613                                    <1> CODEC_HP_VOL_REG               equ     04h  ; headphone volume ; AD1980
3614                                    <1> CODEC_MASTER_MONO_VOL_REG      equ     06h  ; master mono volume (mono-out)
3615                                    <1> ;CODEC_MASTER_TONE_REG         equ     08h  ; master tone (R+L) ; (not used)
3616                                    <1> CODEC_PCBEEP_VOL_REG           equ     0Ah  ; PC beep volume ; ALC655
3617                                    <1> CODEC_PHONE_VOL_REG            equ     0Ch  ; phone volume
3618                                    <1> CODEC_MIC_VOL_REG              equ     0Eh  ; mic volume
3619                                    <1> CODEC_LINE_IN_VOL_REG          equ     10h  ; line in volume
3620                                    <1> CODEC_CD_VOL_REG               equ     12h  ; CD volume
3621                                    <1> ;CODEC_VID_VOL_REG             equ     14h  ; video volume ; (not used)
3622                                    <1> CODEC_AUX_VOL_REG              equ     16h  ; aux volume
3623                                    <1> CODEC_PCM_OUT_REG              equ     18h  ; PCM out volume
3624                                    <1> CODEC_RECORD_SELECT_REG        equ     1Ah  ; record select
3625                                    <1> CODEC_RECORD_VOL_REG           equ     1Ch  ; record volume (record gain)
3626                                    <1> ;CODEC_RECORD_MIC_VOL_REG      equ     1Eh  ; record mic volume ; (not used)
3627                                    <1> CODEC_GP_REG                   equ     20h  ; general purpose
3628                                    <1> ;CODEC_3D_CONTROL_REG          equ     22h  ; 3D control
3629                                    <1> ;;CODEC_AUDIO_INT_PAGING_REG   equ    24h   ; audio int & paging ; (not used)
3630                                    <1> CODEC_POWER_CTRL_REG           equ     26h  ; power down control
3631                                    <1> CODEC_EXT_AUDIO_REG            equ     28h  ; extended audio ID
3632                                    <1> CODEC_EXT_AUDIO_CTRL_REG       equ     2Ah  ; extended audio status/control
3633                                    <1> CODEC_PCM_FRONT_DACRATE_REG    equ     2Ch  ; PCM front sample rate
3634                                    <1> CODEC_PCM_SURND_DACRATE_REG    equ     2Eh  ; PCM surround sample rate
3635                                    <1> CODEC_PCM_LFE_DACRATE_REG      equ     30h  ; PCM Center/LFE sample rate
3636                                    <1> CODEC_LR_ADCRATE_REG           equ     32h  ; PCM input sample rate
3637                                    <1> CODEC_MIC_ADCRATE_REG          equ     34h  ; mic in sample rate  ; AD1980
3638                                    <1> CODEC_PCM_LFE_VOL_REG          equ     36h  ; PCM Center/LFE volume
3639                                    <1> CODEC_PCM_SURND_VOL_REG        equ     38h  ; PCM surround volume
3640                                    <1> ;CODEC_SPDIF_CTRL_REG          equ     3Ah  ; S/PDIF control
3641                                    <1> ; 22/07/2020
3642                                    <1> CODEC_MISC_CRTL_BITS_REG  equ    76h    ; misc control bits ; AD1980
3643                                    <1> ;
3644                                    <1> CODEC_VENDOR_ID1          equ    7Ch    ; REALTEK: 414Ch, ADI: 4144h
3645                                    <1> CODEC_VENDOR_ID2          equ    7Eh    ; REALTEK: 4760h, ADI: 5370h
3646                                    <1>
3647                                    <1> ; VT8233 SGD bits (21/04/2017)
3648                                    <1> FLAG  EQU BIT30
3649                                    <1> EOL   EQU BIT31
3650                                    <1>
3651                                    <1> ; INTEL ICH EQUATES
3652                                    <1> ; 28/05/2017
3653                                    <1> INTEL_VID    equ    8086h  ; Intel's PCI vendor ID
3654                                    <1> ICH_DID      equ    2415h  ; ICH (82801AA) device ID
3655                                    <1> NAMBAR_REG   equ    10h    ; native audio mixer Base Address Register
3656                                    <1> NABMBAR_REG  equ    14h    ; native audio bus mastering Base Addr Reg
3657                                    <1>
3658                                    <1> PI_CR_REG    equ     0Bh     ; PCM in Control Register
3659                                    <1> PO_CR_REG    equ     1Bh     ; PCM out Control Register
3660                                    <1> MC_CR_REG    equ     2Bh     ; MIC in Control Register
3661                                    <1>
3662                                    <1> PI_SR_REG    equ     6       ; PCM in Status register
3663                                    <1> PO_SR_REG    equ     16h     ; PCM out Status register
3664                                    <1> MC_SR_REG    equ     26h     ; MIC in Status register
3665                                    <1>
3666                                    <1> IOCE         equ     BIT4    ; interrupt on complete enable.
3667                                    <1> FEIFE        equ     BIT3    ; set if you want an interrupt to fire
3668                                    <1> LVBIE        equ     BIT2    ; last valid buffer interrupt enable.
3669                                    <1> RR           equ     BIT1    ; reset registers. Nukes all regs
3670                                    <1>                                 ; except bits 4:2 of this register.
3671                                    <1>                                 ; Only set this bit if BIT 0 is 0
3672                                    <1> RPBM         equ     BIT0    ; Run/Pause
3673                                    <1>                              ; set this bit to start the codec!
3674                                    <1>
3675                                    <1> PI_BDBAR_REG equ     0       ; PCM in buffer descriptor BAR
3676                                    <1> PO_BDBAR_REG equ     10h     ; PCM out buffer descriptor BAR
3677                                    <1> MC_BDBAR_REG equ     20h     ; MIC in buffer descriptor BAR
3678                                    <1>
3679                                    <1> PI_CIV_REG   equ     4       ; PCM in current Index value (RO)
3680                                    <1> PO_CIV_REG   equ     14h     ; PCM out current Index value (RO)
3681                                    <1> MC_CIV_REG   equ     24h     ; MIC in current Index value (RO)
3682                                    <1>
3683                                    <1> PI_LVI_REG   equ     5       ; PCM in Last Valid Index
3684                                    <1> PO_LVI_REG   equ     15h     ; PCM out Last Valid Index
3685                                    <1> MC_LVI_REG   equ     25h     ; MIC in Last Valid Index
3686                                    <1>
3687                                    <1> IOC          equ     BIT31 ; Fire an interrupt whenever this
3688                                    <1>                            ; buffer is complete.
```

```
                        <1> BUP   equ     BIT30       ; Buffer Underrun Policy.
                        <1>
                        <1> GLOB_CNT_REG  equ     2Ch ; Global Control Register
                        <1> GLOB_STS_REG equ     30h  ; Global Status register (RO)
                        <1>
                        <1> CTRL_ST_CREADY     equ   BIT8+BIT9+BIT28 ; Primary Codec Ready
                        <1>
                        <1> CODEC_REG_POWERDOWN   equ 26h
                        <1> CODEC_REG_ST          equ 26h
                        <1>
                        <1> ; 22/06/2017
                        <1> PO_PICB_REG  equ 18h      ; PCM Out Position In Current Buffer Register
                        <1>
                        <1> ;============================================================================
                        <1> ;               CODE
                        <1> ;============================================================================
                        <1>
                        <1> ; CODE for INTEL ICH AC'97 AUDIO CONTROLLER
                        <1>
                        <1> DetectICH:
                        <1>       ; 10/06/2017
                        <1>       ; 05/06/2017
                        <1>       ; 29/05/2017
                        <1>       ; 28/05/2017
00014448 B886801524     <1>       mov     eax, (ICH_DID << 16) + INTEL_VID
0001444D E876000000     <1>       call    pciFindDevice
00014452 730D           <1>       jnc     short d_ac97_1
                        <1> d_ac97_0:
                        <1> ; couldn't find the audio device!
00014454 C3             <1>       retn
                        <1>
                        <1> ; CODE for VIA VT8233 AUDIO CONTROLLER
                        <1>
                        <1> DetectVT8233:
                        <1>       ; 10/06/2017
                        <1>       ; 05/06/2017
                        <1>       ; 29/05/2017
                        <1>       ; 03/04/2017
00014455 B806115930     <1>       mov     eax, (VT8233_DID << 16) + VIA_VID
0001445A E869000000     <1>       call    pciFindDevice
                        <1> ;     jnc     short d_vt8233_0
                        <1> ; couldn't find the audio device!
                        <1> ;     retn
0001445F 72F3           <1>       jc    short d_ac97_0  ; 28/05/2017
                        <1> d_vt8233_0:
                        <1>       ; 24/03/2017 ('player.asm')
                        <1>       ; 12/11/2016
                        <1>       ; Erdogan Tan - 8/11/2016
                        <1>       ; References: Kolibrios - vt823x.asm (2016)
                        <1>       ;           VIA VT8235 V-Link South Bridge (VT8235-VIA.PDF)(2002)
                        <1>       ;           lowlevel.eu - AC97 (2016)
                        <1>       ;           .wav player for DOS by Jeff Leyda (2002) -this file-
                        <1>       ;           Linux kernel - via82xx.c (2016)
                        <1> d_ac97_1:
                        <1>       ; eax = BUS/DEV/FN
                        <1>       ;     00000000BBBBBBBBDDDDDFFF00000000
                        <1>       ; edx = DEV/VENDOR
                        <1>       ;     DDDDDDDDDDDDDDDDVVVVVVVVVVVVVVVV
                        <1>
00014461 A3[E08F0100]   <1>       mov   [audio_dev_id], eax
00014466 8915[E48F0100] <1>       mov   [audio_vendor], edx
                        <1>
                        <1>       ; init controller
0001446C B004           <1>       mov   al, PCI_CMD_REG ; command register (04h)
0001446E E8E2000000     <1>       call  pciRegRead32
                        <1>
                        <1>       ; eax = BUS/DEV/FN/REG
                        <1>       ; edx = STATUS/COMMAND
                        <1>       ;     SSSSSSSSSSSSSSSSCCCCCCCCCCCCCCCC
00014473 8915[E88F0100] <1>       mov   [audio_stats_cmd], edx
                        <1>
00014479 B010           <1>       mov   al, PCI_IO_BASE ; IO base address register (10h)
                        <1>       ;mov  al, NAMBAR_REG     ; Native Audio Mixer BAR (10h)
0001447B E8D5000000     <1>       call  pciRegRead32
                        <1>
00014480 66813D[E48F0100]86- <1>  cmp   word [audio_vendor], 8086h ; AC'97 ?
00014488 80             <1>
00014489 751F           <1>       jne   short d_vt8233_1
                        <1>
0001448B 6683E2FE       <1>       and   dx, 0FFFEh ; Audio Codec IO_ADDR_MASK
0001448F 668915[10900100] <1>     mov   [NAMBAR], dx
                        <1>
00014496 B014           <1>       mov   al, NABMBAR_REG ; Native Audio Bus Mastering BAR (14h)
00014498 E8B8000000     <1>       call  pciRegRead32
                        <1>
0001449D 6683E2C0       <1>       and   dx, 0FFC0h ; Audio Controller IO_ADDR_MASK
000144A1 668915[12900100] <1>     mov   [NABMBAR], dx
                        <1>       ;mov  [audio_io_base], dx
                        <1>
000144A8 EB0B           <1>       jmp   short d_ac97_2
                        <1>
                        <1> d_vt8233_1:
000144AA 6683E2C0       <1>       and     dx, 0FFC0h ; Audio Controller IO_ADDR_MASK
000144AE 668915[DE8F0100] <1>     mov     [audio_io_base], dx
                        <1>
                        <1> d_ac97_2:
                        <1>       ; 10/06/2017
000144B5 B03C           <1>       mov   al, AC97_INT_LINE ; Interrupt Line Register (3Ch)
                        <1>       ;call pciRegRead32
000144B7 E886000000     <1>       call  pciRegRead8
                        <1>
                        <1>       ;and  edx, 0FFh
000144BC 6681E2FF00     <1>       and   dx, 0FFh
                        <1>
000144C1 8815[DB8F0100] <1>       mov   [audio_intr], dl
```

```
3793                                    <1>
3794 000144C7 C3                       <1>       retn
3795                                    <1>
3796                                    <1>       ;; (Note: Interrupts are already enabled by TRDOS 386 kernel!)
3797                                    <1>       ;mov   cx, dx
3798                                    <1>
3799                                    <1>       ;in    al, 0A1h ; irq 8-15
3800                                    <1>       ;mov   ah, al
3801                                    <1>       ;in    al, 21h  ; irq 0-7
3802                                    <1>       ;btr   ax, dx  ; unmask ; 17/03/2017
3803                                    <1>       ;;bts  ax, dx   ; MASK interrupt ; 10/06/2017
3804                                    <1>       ;out   21h, al  ; irq <= 7
3805                                    <1>       ;mov   al, ah
3806                                    <1>       ;out   0A1h, al ; irq > 7
3807                                    <1>       ;
3808                                    <1>
3809                                    <1>       ; 10/06/2017
3810                                    <1>       ; === Intel ICH I/O Controller Hub Datasheet, Section 8.1.16 ===
3811                                    <1>       ; PRQ[n]_ROUT Register (61h, PRQB) Bit 7:
3812                                    <1>       ; Interrupt Routing Enable (IRQEN).
3813                                    <1>       ; 0 = The corresponding PIRQ is routed to one of the ISA-compatible
3814                                    <1>       ;     interrupts specified in bits[3:0].
3815                                    <1>       ; 1 = The PIRQ is not routed to the 8259.
3816                                    <1>       ; Note: If the PIRQ is intended to cause an interrupt to the ICH's
3817                                    <1>       ;     integrated I/O APIC, then this bit should be set to 0 and
3818                                    <1>       ;     the APIC_EN bit should be set to 1.
3819                                    <1>       ;     The IRQEN must be set to 0 and the PIRQ routed to
3820                                    <1>       ;     an 8259 interrupt via the IRQ Routing filed (bits[3:0).
3821                                    <1>       ;     The corresponding 8259 interrupt must be masked via the
3822                                    <1>       ;     appropriated bit in the 8259's OCW1 (Interrupt Mask)
3823                                    <1>       ;     register. The IOAPIC must then be enabled by setting
3824                                    <1>       ;     the APIC_EN bit in the GEN_CNTL register.
3825                                    <1>
3826                                    <1>       ;mov   eax, 0F861h  ; D31:F0
3827                                    <1>             ;AL=61h : PIRQ[B] Routing Control Reg, LPC interface
3828                                    <1>       ;;mov  dl, [audio_intr]
3829                                    <1>       ;call  pciRegWrite8
3830                                    <1>       ;;mov  al, 0D0h ; General Control Register (GEN_CTL)
3831                                    <1>       ;;call pciRegRead32
3832                                    <1>       ;;or   edx, 100h ; Bit 8, APIC_EN (Enable I/O APIC)
3833                                    <1>       ;;;call      pciRegWrite32
3834                                    <1>       ;;and  edx, ~100h
3835                                    <1>       ;;call pciRegWrite32 ; ; Bit 8, APIC_EN (Disable I/O APIC)
3836                                    <1>       ;
3837                                    <1>
3838                                    <1>       ;mov   dx, 4D1h     ; 8259 ELCR2
3839                                    <1>       ;in    al, dx
3840                                    <1>       ;mov   ah, al
3841                                    <1>       ;;mov  dx, 4D0h     ; 8259 ELCR1
3842                                    <1>       ;dec   dl
3843                                    <1>       ;in    al, dx
3844                                    <1>       ;bts   ax, cx
3845                                    <1>       ;;mov  dx, 4D0h
3846                                    <1>       ;out   dx, al       ; set level-triggered mode
3847                                    <1>       ;mov   al, ah ; 29/05/2017
3848                                    <1>       ;;mov  dx, 4D1h
3849                                    <1>       ;inc   dl
3850                                    <1>       ;out   dx, al       ; set level-triggered mode
3851                                    <1>
3852                                    <1>       ;xor   eax, eax ; 0
3853                                    <1>
3854                                    <1>       ;retn
3855                                    <1>
3856                                    <1> ; CODE for PCI
3857                                    <1>
3858                                    <1> pciFindDevice:
3859                                    <1>       ; 03/04/2017 ('pci.asm', 20/03/2017)
3860                                    <1>       ;
3861                                    <1>       ; scan through PCI space looking for a device+vendor ID
3862                                    <1>       ;
3863                                    <1>       ; Entry: EAX=Device+Vendor ID
3864                                    <1>       ;
3865                                    <1>       ; Exit: EAX=PCI address if device found
3866                                    <1>       ;       EDX=Device+Vendor ID
3867                                    <1>       ;        CY clear if found, set if not found. EAX invalid if CY set.
3868                                    <1>       ;
3869                                    <1>       ; Destroys: ebx, esi, edi, cl
3870                                    <1>       ;
3871                                    <1>
3872                                    <1>       ;push  ecx
3873 000144C8 50                       <1>       push   eax
3874                                    <1>       ;push  esi
3875                                    <1>       ;push  edi
3876                                    <1>
3877 000144C9 89C6                     <1>       mov    esi, eax                 ; save off vend+device ID
3878 000144CB BF00FFFF7F               <1>       mov    edi, (80000000h - 100h) ; start with bus 0, dev 0 func 0
3879                                    <1>
3880                                    <1> nextPCIdevice:
3881 000144D0 81C700010000             <1>       add    edi, 100h
3882 000144D6 81FF00F8FF80             <1>       cmp    edi, 80FFF800h            ; scanned all devices?
3883 000144DC F9                       <1>       stc
3884 000144DD 740C                     <1>       je     short PCIScanExit       ; not found
3885                                    <1>
3886 000144DF 89F8                     <1>       mov    eax, edi                 ; read PCI registers
3887 000144E1 E86F000000               <1>       call   pciRegRead32
3888 000144E6 39F2                     <1>       cmp    edx, esi                 ; found device?
3889 000144E8 75E6                     <1>       jne    short nextPCIdevice
3890 000144EA F8                       <1>       clc
3891                                    <1>
3892                                    <1> PCIScanExit:
3893 000144EB 9C                       <1>       pushf
3894 000144EC B8FFFFFF7F               <1>       mov    eax, NOT_BIT31      ; 19/03/2017
3895 000144F1 21F8                     <1>       and    eax, edi     ; return only bus/dev/fn #
3896 000144F3 9D                       <1>       popf
3897                                    <1>
```

```
                                <1>       ;pop   edi
                                <1>       ;pop   esi
000144F4 5A                     <1>       pop    edx
                                <1>       ;pop   ecx
000144F5 C3                     <1>       retn
                                <1>
                                <1> pciRegRead:
                                <1>       ; 03/04/2017 ('pci.asm', 20/03/2017)
                                <1>       ;
                                <1>       ; 8/16/32bit PCI reader
                                <1>       ;
                                <1>       ; Entry: EAX=PCI Bus/Device/fn/register number
                                <1>       ;           BIT30 set if 32 bit access requested
                                <1>       ;           BIT29 set if 16 bit access requested
                                <1>       ;           otherwise defaults to 8 bit read
                                <1>       ;
                                <1>       ; Exit:  DL,DX,EDX register data depending on requested read size
                                <1>       ;
                                <1>       ; Note1: this routine is meant to be called via pciRegRead8,
                                <1>       ;      pciRegread16 or pciRegRead32, listed below.
                                <1>       ;
                                <1>       ; Note2: don't attempt to read 32 bits of data from a non dword
                                <1>       ;      aligned reg number. Likewise, don't do 16 bit reads from
                                <1>       ;      non word aligned reg #
                                <1>
000144F6 53                     <1>       push   ebx
000144F7 51                     <1>       push   ecx
000144F8 89C3                   <1>       mov    ebx, eax           ; save eax, dh
000144FA 88F1                   <1>       mov    cl, dh
                                <1>
000144FC 25FFFFFF3F             <1>       and   eax, NOT_PCI32_PCI16; clear out data size request
00014501 0D00000080             <1>       or    eax, BIT31          ; make a PCI access request
00014506 24FC                   <1>       and   al, ~3 ; NOT 3      ; force index to be dword
                                <1>
00014508 66BAF80C               <1>       mov    dx, PCI_INDEX_PORT
0001450C EF                     <1>       out    dx, eax            ; write PCI selector
                                <1>
0001450D 66BAFC0C               <1>       mov   dx, PCI_DATA_PORT
00014511 88D8                   <1>       mov   al, bl
00014513 2403                   <1>       and   al, 3               ; figure out which port to
00014515 00C2                   <1>       add   dl, al              ; read to
                                <1>
00014517 F7C3000000C0           <1>       test   ebx, PCI32+PCI16
0001451D 7507                   <1>       jnz   short _pregr0
0001451F EC                     <1>       in     al, dx             ; return 8 bits of data
00014520 88C2                   <1>       mov    dl, al
00014522 88CE                   <1>       mov    dh, cl             ; restore dh for 8 bit read
00014524 EB12                   <1>       jmp    short _pregr2
                                <1> _pregr0:
00014526 F7C300000080           <1>       test   ebx, PCI32
0001452C 7507                   <1>       jnz    short _pregr1
0001452E 66ED                   <1>       in     ax, dx
00014530 6689C2                 <1>       mov    dx, ax             ; return 16 bits of data
00014533 EB03                   <1>       jmp    short _pregr2
                                <1> _pregr1:
00014535 ED                     <1>       in     eax, dx            ; return 32 bits of data
00014536 89C2                   <1>       mov    edx, eax
                                <1> _pregr2:
00014538 89D8                   <1>       mov    eax, ebx           ; restore eax
0001453A 25FFFFFF3F             <1>       and    eax, NOT_PCI32_PCI16  ; clear out data size request
0001453F 59                     <1>       pop    ecx
00014540 5B                     <1>       pop    ebx
00014541 C3                     <1>       retn
                                <1>
                                <1> pciRegRead8:
00014542 25FFFFFF3F             <1>       and    eax, NOT_PCI32_PCI16 ; set up 8 bit read size
00014547 EBAD                   <1>       jmp    short pciRegRead    ; call generic PCI access
                                <1>
                                <1> pciRegRead16:
00014549 25FFFFFF3F             <1>       and    eax, NOT_PCI32_PCI16 ; set up 16 bit read size
0001454E 0D00000040             <1>       or     eax, PCI16          ; call generic PCI access
00014553 EBA1                   <1>       jmp    short pciRegRead
                                <1>
                                <1> pciRegRead32:
00014555 25FFFFFF3F             <1>       and    eax, NOT_PCI32_PCI16 ; set up 32 bit read size
0001455A 0D00000080             <1>       or     eax, PCI32          ; call generic PCI access
0001455F EB95                   <1>       jmp    pciRegRead
                                <1>
                                <1> pciRegWrite:
                                <1>       ; 03/04/2017 ('pci.asm', 29/11/2016)
                                <1>       ;
                                <1>       ; 8/16/32bit PCI writer
                                <1>       ;
                                <1>       ; Entry: EAX=PCI Bus/Device/fn/register number
                                <1>       ;           BIT31 set if 32 bit access requested
                                <1>       ;           BIT30 set if 16 bit access requested
                                <1>       ;           otherwise defaults to 8bit read
                                <1>       ;        DL/DX/EDX data to write depending on size
                                <1>       ;
                                <1>       ; Note1: this routine is meant to be called via pciRegWrite8,
                                <1>       ;      pciRegWrite16 or pciRegWrite32 as detailed below.
                                <1>       ;
                                <1>       ; Note2: don't attempt to write 32bits of data from a non dword
                                <1>       ;      aligned reg number. Likewise, don't do 16 bit writes from
                                <1>       ;      non word aligned reg #
                                <1>
00014561 53                     <1>       push   ebx
00014562 51                     <1>       push   ecx
00014563 89C3                   <1>       mov    ebx, eax           ; save eax, edx
00014565 89D1                   <1>       mov    ecx, edx
00014567 25FFFFFF3F             <1>       and    eax, NOT_PCI32_PCI16 ; clear out data size request
0001456C 0D00000080             <1>       or     eax, BIT31          ; make a PCI access request
00014571 24FC                   <1>       and    al, ~3 ; NOT 3      ; force index to be dword
                                <1>
00014573 66BAF80C               <1>       mov    dx, PCI_INDEX_PORT
```

```
4003 00014577 EF                <1>       out    dx, eax             ; write PCI selector
4004                            <1>
4005 00014578 66BAFC0C          <1>       mov    dx, PCI_DATA_PORT
4006 0001457C 88D8              <1>       mov    al, bl
4007 0001457E 2403              <1>       and    al, 3               ; figure out which port to
4008 00014580 00C2              <1>       add    dl, al              ; write to
4009                            <1>
4010 00014582 F7C3000000C0      <1>       test   ebx, PCI32+PCI16
4011 00014588 7505              <1>       jnz    short _pregw0
4012 0001458A 88C8              <1>       mov    al, cl              ; put data into al
4013 0001458C EE                <1>       out    dx, al
4014 0001458D EB12              <1>       jmp    short _pregw2
4015                            <1> _pregw0:
4016 0001458F F7C300000080      <1>       test   ebx, PCI32
4017 00014595 7507              <1>       jnz    short _pregw1
4018 00014597 6689C8            <1>       mov    ax, cx              ; put data into ax
4019 0001459A 66EF              <1>       out    dx, ax
4020 0001459C EB03              <1>       jmp    short _pregw2
4021                            <1> _pregw1:
4022 0001459E 89C8              <1>       mov    eax, ecx            ; put data into eax
4023 000145A0 EF                <1>       out    dx, eax
4024                            <1> _pregw2:
4025 000145A1 89D8              <1>       mov    eax, ebx            ; restore eax
4026 000145A3 25FFFFFF3F        <1>       and    eax, NOT_PCI32_PCI16 ; clear out data size request
4027 000145A8 89CA              <1>       mov    edx, ecx            ; restore dx
4028 000145AA 59                <1>       pop    ecx
4029 000145AB 5B                <1>       pop    ebx
4030 000145AC C3                <1>       retn
4031                            <1>
4032                            <1> pciRegWrite8:
4033 000145AD 25FFFFFF3F        <1>       and    eax, NOT_PCI32_PCI16 ; set up 8 bit write size
4034 000145B2 EBAD              <1>       jmp    short pciRegWrite   ; call generic PCI access
4035                            <1>
4036                            <1> pciRegWrite16:
4037 000145B4 25FFFFFF3F        <1>       and    eax, NOT_PCI32_PCI16 ; set up 16 bit write size
4038 000145B9 0D00000040        <1>       or     eax, PCI16          ; call generic PCI access
4039 000145BE EBA1              <1>       jmp    short pciRegWrite
4040                            <1>
4041                            <1> pciRegWrite32:
4042 000145C0 25FFFFFF3F        <1>         and     eax, NOT_PCI32_PCI16 ; set up 32 bit write size
4043 000145C5 0D00000080        <1>         or      eax, PCI32      ; call generic PCI access
4044 000145CA EB95              <1>         jmp  pciRegWrite
4045                            <1>
4046                            <1> init_codec:
4047                            <1>       ; 05/06/2017
4048                            <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
4049                            <1>       ;
4050 000145CC A1[E08F0100]      <1>       mov    eax, [audio_dev_id]
4051 000145D1 B041              <1>       mov    al, VIA_ACLINK_CTRL
4052 000145D3 E86AFFFFFF        <1>       call   pciRegRead8
4053                            <1>       ; ?
4054 000145D8 B040              <1>       mov    al, VIA_ACLINK_STAT
4055 000145DA E863FFFFFF        <1>       call   pciRegRead8
4056 000145DF F6C201            <1>       test   dl, VIA_ACLINK_C00_READY
4057 000145E2 7508              <1>         jnz     short _codec_ready_1
4058 000145E4 E80E000000        <1>       call   reset_codec
4059 000145E9 7306              <1>       jnc    short _codec_ready_2 ; eax = 1
4060 000145EB C3                <1>       retn
4061                            <1> _codec_ready_1:
4062 000145EC B801000000        <1>       mov    eax, 1
4063                            <1> _codec_ready_2:
4064 000145F1 E886000000        <1>       call   codec_io_w16
4065                            <1> detect_codec:
4066 000145F6 C3                <1>       retn
4067                            <1>
4068                            <1> reset_codec:
4069                            <1>       ; 16/04/2017
4070                            <1>       ; 23/03/2017
4071                            <1>       ; ('codec.asm')
4072                            <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
4073 000145F7 A1[E08F0100]      <1>       mov    eax, [audio_dev_id]
4074 000145FC B041              <1>       mov    al, VIA_ACLINK_CTRL
4075 000145FE B2E0              <1>       mov    dl, VIA_ACLINK_CTRL_ENABLE + VIA_ACLINK_CTRL_RESET + VIA_ACLINK_CTRL_SYNC
4076 00014600 E8A8FFFFFF        <1>       call   pciRegWrite8
4077                            <1>
4078 00014605 E849000000        <1>       call   delay_100ms  ; wait 100 ms
4079                            <1> _rc_cold:
4080 0001460A E814000000        <1>       call   cold_reset
4081 0001460F 7301              <1>       jnc    short _reset_codec_ok
4082                            <1>
4083                            <1>       ; 16/04/2017
4084                            <1>       ;xor   eax, eax     ; timeout error
4085                            <1>       ;stc
4086 00014611 C3                <1>       retn
4087                            <1>
4088                            <1> _reset_codec_ok:
4089                            <1>       ; 01/09/2020
4090                            <1>       ; 15/08/2020
4091                            <1>       ; 27/07/2020
4092                            <1>       ; also reset codec by using index control register 0 of AD1980 or ALC655
4093                            <1>       ; (to fix line out -2 channels audio playing- problem on AD1980 codec)
4094                            <1>
4095 00014612 29C0              <1>       sub    eax, eax
4096 00014614 BA00000000        <1>       mov    edx, CODEC_RESET_REG ; 00h ; Reset register
4097 00014619 E8CA000000        <1>       call   codec_write
4098                            <1>
4099                            <1>       ;sub   eax, eax
4100                            <1>       ; 01/09/2020
4101                            <1>       ; 15/08/2020
4102                            <1>       ; AD1980 BugFix
4103                            <1>       ; (set HPSEL -headphone amp to be driven from mixer- and
4104                            <1>       ;      CLDIS - center and LFE disable- bits)
4105                            <1>       ;mov   eax, 0C00h ; HPSEL = bit 10, CLDIS = bit 11 ; 01/09/2020
4106                            <1>       ;mov   edx, CODEC_MISC_CRTL_BITS_REG ; 76h ; Misc Ctrl Bits ; AD1980
4107                            <1>       ;call  codec_write
```

```
                                <1>
0001461E 31C0                   <1>       xor     eax, eax
                                <1>       ;mov   al, VIA_ACLINK_C00_READY ; 1
00014620 FEC0                   <1>       inc    al
00014622 C3                     <1>       retn
                                <1>
                                <1> cold_reset:
                                <1>       ; 16/04/2017
                                <1>       ; 23/03/2017
                                <1>       ; ('codec.asm')
                                <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
                                <1>       ;mov   eax, [audio_dev_id]
                                <1>       ;mov   al, VIA_ACLINK_CTRL
00014623 30D2                   <1>       xor    dl, dl ; 0
00014625 E883FFFFFF             <1>       call   pciRegWrite8
                                <1>
0001462A E824000000             <1>       call   delay_100ms  ; wait 100 ms
                                <1>
                                <1>       ;; ACLink on, deassert ACLink reset, VSR, SGD data out
                                <1>       ;; note - FM data out has trouble with non VRA codecs !!
                                <1>
                                <1>       ;mov   eax, [audio_dev_id]
                                <1>       ;mov   al, VIA_ACLINK_CTRL
0001462F B2CC                   <1>       mov    dl, VIA_ACLINK_CTRL_INIT
00014631 E877FFFFFF             <1>       call   pciRegWrite8
                                <1>
00014636 B910000000             <1>       mov    ecx, 16      ; total 2s
                                <1>
                                <1> _crst_wait:
                                <1>       ;mov   eax, [audio_dev_id]
0001463B B040                   <1>       mov    al, VIA_ACLINK_STAT
0001463D E800FFFFFF             <1>       call   pciRegRead8
                                <1>
00014642 F6C201                 <1>       test   dl, VIA_ACLINK_C00_READY
00014645 750B                   <1>       jnz    short _crst_ok
                                <1>
00014647 51                     <1>       push   ecx
00014648 E806000000             <1>       call   delay_100ms
0001464D 59                     <1>       pop    ecx
                                <1>
0001464E 49                     <1>       dec    ecx
0001464F 75EA                   <1>       jnz    short _crst_wait
                                <1>
                                <1> _crst_fail:
00014651 F9                     <1>       stc
                                <1> _crst_ok:
00014652 C3                     <1>       retn
                                <1>
                                <1> delay_100ms:
                                <1>       ; 29/05/2017
                                <1>       ; 24/03/2017 ('codec.asm')
                                <1>       ; wait 100 ms
00014653 B990010000             <1>       mov    ecx, 400  ; 400*0.25ms
                                <1> _delay_x_ms:
00014658 E803000000             <1>       call   delay1_4ms
0001465D E2F9                   <1>       loop   _delay_x_ms
0001465F C3                     <1>       retn
                                <1>
                                <1> ;     delay1_4ms - Delay for 1/4 millisecond.
                                <1> ;         1mS = 1000us
                                <1> ;       Entry:
                                <1> ;         None
                                <1> ;       Exit:
                                <1> ;       None
                                <1> ;
                                <1> ;       Modified:
                                <1> ;         None
                                <1> ;
                                <1>
                                <1>       ; 29/05/2017
                                <1>       ; 23/04/2017
                                <1>       ; 05/03/2017 (TRDOS 386)
                                <1>       ; ('UTILS.ASM')
                                <1> delay1_4ms:
00014660 50                     <1>       push   eax
00014661 51                     <1>       push   ecx
00014662 B110                   <1>       mov    cl, 16       ; close enough.
                                <1>
00014664 E461                   <1>       in     al, PORTB ; 61h
                                <1>
00014666 2410                   <1>       and    al, REFRESH_STATUS ; 10h
00014668 88C5                   <1>       mov    ch, al       ; Start toggle state
                                <1> _d4ms1:
0001466A E461                   <1>       in     al, PORTB    ; Read system control port
                                <1>
0001466C 2410                   <1>       and    al, REFRESH_STATUS ; Refresh toggles 15.085 microseconds
0001466E 38C5                   <1>       cmp    ch, al
00014670 74F8                   <1>       je     short _d4ms1 ; Wait for state change
                                <1>
00014672 88C5                   <1>       mov    ch, al       ; Update with new state
00014674 FEC9                   <1>       dec    cl
00014676 75F2                   <1>       jnz    short _d4ms1
                                <1>
00014678 F8                     <1>       clc    ; 29/05/2017
                                <1>
00014679 59                     <1>       pop    ecx
0001467A 58                     <1>       pop    eax
0001467B C3                     <1>       retn
                                <1>
                                <1> ; 10/04/2017 (TRDOS 386)
                                <1> ; 12/11/2016
                                <1>
                                <1> codec_io_w16: ;w32
                                <1>       ; ('codec.asm')
0001467C 668B15[DE8F0100]       <1>       mov    dx, [audio_io_base]
```

```
4213 00014683 6681C28000          <1>       add    dx, VIA_REG_AC97
4214 00014688 EF                  <1>       out    dx, eax
4215 00014689 C3                  <1>       retn
4216                              <1>
4217                              <1> codec_io_r16: ;r32
4218                              <1>       ; ('codec.asm')
4219 0001468A 668B15[DE8F0100]    <1>       mov    dx, [audio_io_base]
4220 00014691 6681C28000          <1>       add    dx, VIA_REG_AC97
4221 00014696 ED                  <1>       in     eax, dx
4222 00014697 C3                  <1>       retn
4223                              <1>
4224                              <1> ctrl_io_w8:
4225                              <1>       ; ('codec.asm')
4226 00014698 660315[DE8F0100]    <1>       add    dx, [audio_io_base]
4227 0001469F EE                  <1>       out    dx, al
4228 000146A0 C3                  <1>       retn
4229                              <1>
4230                              <1> ctrl_io_r8:
4231                              <1>       ; ('codec.asm')
4232 000146A1 660315[DE8F0100]    <1>       add    dx, [audio_io_base]
4233 000146A8 EC                  <1>       in     al, dx
4234 000146A9 C3                  <1>       retn
4235                              <1>
4236                              <1> ctrl_io_w32:
4237                              <1>       ; ('codec.asm')
4238 000146AA 660315[DE8F0100]    <1>       add    dx, [audio_io_base]
4239 000146B1 EF                  <1>       out    dx, eax
4240 000146B2 C3                  <1>       retn
4241                              <1>
4242                              <1> ctrl_io_r32:
4243                              <1>       ; ('codec.asm')
4244 000146B3 660315[DE8F0100]    <1>       add    dx, [audio_io_base]
4245 000146BA ED                  <1>       in     eax, dx
4246 000146BB C3                  <1>       retn
4247                              <1>
4248                              <1> codec_read:
4249                              <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
4250                              <1>       ; Use only primary codec.
4251                              <1>       ; eax = register
4252 000146BC C1E010              <1>       shl    eax, VIA_REG_AC97_CMD_SHIFT
4253 000146BF 0D00008002          <1>       or     eax, VIA_REG_AC97_PRIMARY_VALID + VIA_REG_AC97_READ
4254                              <1>
4255 000146C4 E8B3FFFFFF          <1>       call   codec_io_w16
4256                              <1>
4257                              <1>              ; codec_valid
4258 000146C9 E831000000          <1>       call   codec_check_ready
4259 000146CE 7301                <1>       jnc    short _cr_ok
4260                              <1>
4261 000146D0 C3                  <1>       retn
4262                              <1>
4263                              <1> _cr_ok:
4264                              <1>       ; wait 25 ms
4265 000146D1 B950000000          <1>       mov    ecx, 80 ; (100*0.25 ms)
4266                              <1> _cr_wloop:
4267 000146D6 E885FFFFFF          <1>       call   delay1_4ms
4268 000146DB E2F9                <1>       loop   _cr_wloop
4269                              <1>
4270 000146DD E8A8FFFFFF          <1>       call   codec_io_r16
4271 000146E2 25FFFF0000          <1>       and    eax, 0FFFFh
4272 000146E7 C3                  <1>       retn
4273                              <1>
4274                              <1> codec_write:
4275                              <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
4276                              <1>         ; Use only primary codec.
4277                              <1>
4278                              <1>       ; eax = data (volume)
4279                              <1>       ; edx = register (mixer register)
4280                              <1>
4281 000146E8 C1E210              <1>       shl    edx, VIA_REG_AC97_CMD_SHIFT
4282                              <1>
4283 000146EB C1E000              <1>       shl    eax, VIA_REG_AC97_DATA_SHIFT ; shl eax, 0
4284 000146EE 09C2                <1>       or     edx, eax
4285                              <1>
4286 000146F0 B800000000          <1>       mov    eax, VIA_REG_AC97_CODEC_ID_PRIMARY
4287 000146F5 C1E01E              <1>       shl    eax, VIA_REG_AC97_CODEC_ID_SHIFT
4288 000146F8 09D0                <1>       or     eax, edx
4289                              <1>
4290 000146FA E87DFFFFFF          <1>       call   codec_io_w16
4291                              <1>       ;mov   [codec.regs+esi], ax
4292                              <1>
4293                              <1>       ;call  codec_check_ready
4294                              <1>       ;retn
4295                              <1>       ;jmp   short _codec_check_ready
4296                              <1>
4297                              <1> codec_check_ready:
4298                              <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
4299                              <1>
4300                              <1> _codec_check_ready:
4301 000146FF B914000000          <1>       mov    ecx, 20    ; total 2s
4302                              <1> _ccr_wait:
4303 00014704 51                  <1>       push   ecx
4304                              <1>
4305 00014705 E880FFFFFF          <1>       call   codec_io_r16
4306 0001470A A900000001          <1>       test   eax, VIA_REG_AC97_BUSY
4307 0001470F 740B                <1>       jz     short _ccr_ok
4308                              <1>
4309 00014711 E83DFFFFFF          <1>       call   delay_100ms
4310                              <1>
4311 00014716 59                  <1>       pop    ecx
4312                              <1>
4313 00014717 49                  <1>       dec    ecx
4314 00014718 75EA                <1>       jnz    short _ccr_wait
4315                              <1>
4316 0001471A F9                  <1>       stc
4317 0001471B C3                  <1>       retn
```

```
                                <1>
                                <1> _ccr_ok:
0001471C 59                     <1>         pop     ecx
0001471D 25FFFF0000             <1>         and     eax, 0FFFFh
00014722 C3                     <1>         retn
                                <1>
                                <1> codec_config:
                                <1>         ; 10/06/2017
                                <1>         ; 29/05/2017
                                <1>         ; 24/04/2017
                                <1>         ; 21/04/2017
                                <1>         ; 16/04/2017 (TRDOS 386 Kernel)
                                <1>         ; 15/11/2016 ('codec.asm', 'player.com')
                                <1>         ; 14/11/2016
                                <1>         ; 12/11/2016 - Erdogan Tan
                                <1>         ;           (Ref: KolibriOS, 'setup_codec', codec.inc)
                                <1>
00014723 B802020000             <1>         mov     eax, 0202h
00014728 66A3[0E900100]         <1>         mov     [audio_master_volume], ax
0001472E 66B81F1F               <1>         mov     ax, 1F1Fh ; 31,31
00014732 BA02000000             <1>         mov     edx, CODEC_MASTER_VOL_REG ; 02h ; Line Out
00014737 E8ACFFFFFF             <1>         call    codec_write
                                <1>         ;jc     short cconfig_error
                                <1>
                                <1>         ;mov    eax, 0202h
0001473C 66B80202               <1>         mov     ax, 0202h
00014740 BA18000000             <1>         mov     edx, CODEC_PCM_OUT_REG ; 18h ; Wave Output (Stereo)
00014745 E89EFFFFFF             <1>         call    codec_write
                                <1>         ;jc     short cconfig_error
                                <1>
                                <1>         ;mov    eax, 0202h
0001474A 66B80202               <1>         mov     ax, 0202h
0001474E BA04000000             <1>         mov     edx, CODEC_AUX_VOL ; 04h ; CODEC_HP_VOL_REG ; HeadPhone
00014753 E890FFFFFF             <1>         call    codec_write
                                <1>         ;jc     short cconfig_error
                                <1>
                                <1>         ;mov    eax, 08h
                                <1>         ;mov    ax, 08h
00014758 66B80880               <1>         mov     ax, 8008h ; Mute
0001475C BA0C000000             <1>         mov     edx, 0Ch  ; AC97_PHONE_VOL ; TAD Input (Mono)
00014761 E882FFFFFF             <1>         call    codec_write
                                <1>         ;jc     short cconfig_error
                                <1>
                                <1>         ;mov    eax, 0808h
00014766 66B80808               <1>         mov     ax, 0808h
0001476A BA10000000             <1>         mov     edx, CODEC_LINE_IN_VOL_REG ; 10h ; Line Input (Stereo)
0001476F E874FFFFFF             <1>         call    codec_write
                                <1>         ;jc     short cconfig_error
                                <1>
                                <1>         ;mov    eax, 0808h
00014774 66B80808               <1>         mov     ax, 0808h
00014778 BA12000000             <1>         mov     edx, CODEC_CD_VOL_REG ; 12h ; CR Input (Stereo)
0001477D E866FFFFFF             <1>         call    codec_write
                                <1>         ;jc     short cconfig_error
                                <1>
                                <1>         ;mov    eax, 0808h
00014782 66B80808               <1>         mov     ax, 0808h
00014786 BA16000000             <1>         mov     edx, CODEC_AUX_VOL_REG ; 16h ; Aux Input (Stereo)
                                <1>         ;call   codec_write
                                <1>         ;;jc    short cconfig_error
0001478B E958FFFFFF             <1>         jmp     codec_write ; 10/06/2017
                                <1>
                                <1> ;       ; Extended Audio Status (2Ah)
                                <1> ;       mov     eax, CODEC_EXT_AUDIO_CTRL_REG ; 2Ah
                                <1> ;       call    codec_read
                                <1> ;       and     eax, 0FFFFh - 2     ; clear DRA (BIT1)
                                <1> ;       ;or     eax, 1              ; set VRA (BIT0)
                                <1> ;       or      eax, 5       ; VRA (BIT0) & S/PDIF (BIT2) ; 14/11/2016
                                <1> ;       mov     edx, CODEC_EXT_AUDIO_CTRL_REG
                                <1> ;       call    codec_write
                                <1> ;       ;jc     short cconfig_error
                                <1> ;
                                <1> ;set_sample_rate:
                                <1> ;       ;movzx eax, word [audio_freq]
                                <1> ;       mov     ax, [audio_freq]
                                <1> ;       mov     edx, CODEC_PCM_FRONT_DACRATE_REG ; 2Ch ; PCM Front DAC Rate
                                <1> ;       ;call   codec_write
                                <1> ;       ;retn
                                <1> ;       jmp     codec_write
                                <1>
                                <1> ;cconfig_error:
                                <1> ;       retn
                                <1>
                                <1> vt8233_int_handler:
                                <1>         ; 27/07/2020
                                <1>         ; 22/07/2020
                                <1>         ; Interrupt Handler for VIA VT8237R Audio Controller
                                <1>         ; Note: called by 'dev_IRQ_service'
                                <1>         ; 14/10/2017
                                <1>         ; 09/10/2017, 10/10/2017, 12/10/2017
                                <1>         ; 13/06/2017
                                <1>         ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
                                <1>         ; 24/03/2017 - 'PLAYER.COM' ('player.asm')
                                <1>
                                <1>         ;push   eax ; * must be saved !
                                <1>         ;push   edx
                                <1>         ;push   ecx
                                <1>         ;push   ebx ; * must be saved !
                                <1>         ;push   esi
                                <1>         ;push   edi
                                <1>
                                <1>         ;cmp    byte [audio_busy], 1
                                <1>         ;jnb    short _ih0 ; 09/10/2017
                                <1>
                                <1>         ;mov    byte [audio_flag_eol], 0
```

```
                                <1>
00014790 66BA0000               <1>       mov   dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STATUS
00014794 E808FFFFFF             <1>       call  ctrl_io_r8
                                <1>
00014799 A880                   <1>       test  al, VIA_REG_STAT_ACTIVE
0001479B 7417                   <1>       jz    short _ih0 ; 09/10/2017
                                <1>
0001479D 2407                   <1>       and   al, VIA_REG_STAT_EOL + VIA_REG_STAT_FLAG + VIA_REG_STAT_STOPPED
0001479F A2[0D900100]           <1>       mov   [audio_flag_eol], al
000147A4 740E                   <1>       jz    short _ih0 ; 09/10/2017
                                <1>
                                <1>       ; 09/10/2017
                                <1>       ;mov  byte [audio_busy], 1
                                <1>
000147A6 803D[0C900100]01       <1>       cmp   byte [audio_play_cmd], 1
000147AD 7315                   <1>       jnb   short _ih1 ; 10/10/2017
                                <1>
000147AF E84A000000             <1>       call  channel_reset
                                <1> _ih0:
                                <1>       ; 09/10/2017
000147B4 A0[0D900100]           <1>       mov   al, [audio_flag_eol]   ;; ack ;;
000147B9 66BA0000               <1>       mov   dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STATUS
000147BD E8D6FEFFFF             <1>       call  ctrl_io_w8
000147C2 EB39                   <1>       jmp   short _ih4
                                <1> _ih1:
                                <1> vt8233_tuneLoop:
000147C4 A0[0D900100]           <1>       mov   al, [audio_flag_eol]   ;; ack ;;
000147C9 66BA0000               <1>       mov   dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STATUS
000147CD E8C6FEFFFF             <1>       call  ctrl_io_w8
                                <1>
                                <1>       ; 22/07/2020
                                <1>       ;; 12/10/2017
                                <1>       ;mov  byte [audio_flag], 0 ; Reset
                                <1>
                                <1>       ; 10/10/2017
                                <1>       ; 09/10/2017
                                <1>       ;test  byte [audio_flag_eol], VIA_REG_STAT_FLAG
                                <1>       ;jz    short _ih2 ; EOL
                                <1>
                                <1>       ; 22/07/2020
                                <1>       ; 14/10/2017
                                <1>       ;test  byte [audio_flag_eol], VIA_REG_STAT_EOL
                                <1>       ;jnz   short _ih2 ; EOL
                                <1>       ;              ; (Half Buffer 2 has been completed
                                <1>       ;              ; and Half Buffer 1 will be played.)
                                <1>
                                <1>       ; FLAG
                                <1>       ; (Half Buffer 1 has been completed
                                <1>       ;  and Half Buffer 2 will be played.)
                                <1>
                                <1>       ; 14/10/2017
                                <1>       ;; (Continue to play.)
                                <1>       ;mov  al, VIA_REG_CTRL_INT
                                <1>       ;or   al, VIA_REG_CTRL_START
                                <1>       ;mov  dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
                                <1>       ;call ctrl_io_w8
                                <1>       ; 12/10/2017
                                <1>       ;mov  byte [audio_flag], 1
                                <1>
                                <1>       ; 22/07/2020
                                <1>       ;inc  byte [audio_flag] ; = 1
                                <1> _ih2:
                                <1>       ; 10/10/2017
000147D2 8B3D[F88F0100]         <1>       mov   edi, [audio_dma_buff]
000147D8 8B0D[FC8F0100]         <1>       mov   ecx, [audio_dmabuff_size]
000147DE D1E9                   <1>       shr   ecx, 1 ; dma buff size / 2 = half buffer size
                                <1>
                                <1>       ; 22/07/2020
                                <1>       ; 12/10/2017
                                <1>       ;cmp  byte [audio_flag], 0
                                <1>       ;ja   short _ih3 ; Playing Half Buffer 2 (Current: FLAG)
                                <1>
                                <1>       ; 27/07/2020
                                <1>       ; 22/07/2020
000147E0 F605[00900100]01       <1>       test  byte [audio_flag], 1  ; Current flag value
000147E7 7402                   <1>       jz    short _ih3 ; Half Buffer 1 must be filled
                                <1>
                                <1>       ; Half Buffer 2 must be filled
000147E9 01CF                   <1>       add   edi, ecx
                                <1> _ih3:
                                <1>       ; Update half buffer 2 while playing half buffer 1
                                <1>       ; Update half buffer 1 while playing half buffer 2
                                <1>
000147EB 8B35[F08F0100]         <1>       mov   esi, [audio_p_buffer] ; phy addr of audio buff
000147F1 C1E902                 <1>       shr   ecx, 2 ; half buff size / 4
000147F4 F3A5                   <1>       rep   movsd
                                <1>
                                <1>       ; switch flag value ;
000147F6 8035[00900100]01       <1>       xor   byte [audio_flag], 1
                                <1>       ; 12/10/2017
                                <1>       ; [audio_flag] = 0 : Playing dma half buffer 2
                                <1>       ;                    ; Next buffer (to update) is dma half buff 1
                                <1>       ;             = 1 : Playing dma half buffer 1
                                <1>       ;                    ; Next buffer (to update) is dma half buff 2
                                <1> _ih4:
                                <1>       ; 28/05/2017
                                <1>       ;mov  byte [audio_busy], 0 ; 09/10/2017
                                <1>       ;
                                <1>       ;pop  edi
                                <1>       ;pop  esi
                                <1>       ;pop  ebx ; * must be restored !
                                <1>       ;pop  ecx
                                <1>       ;pop  edx
                                <1>       ;pop  eax ; * must be restored !
                                <1>
```

```
4528 000147FD C3                <1>       retn
4529                            <1>
4530                            <1> channel_reset:
4531                            <1>       ; 24/06/2017
4532                            <1>       ; 29/05/2017
4533                            <1>       ; 23/03/2017
4534                            <1>       ; 14/11/2016 - Erdogan Tan
4535                            <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
4536 000147FE BA01000000      <1>       mov   edx, VIA_REG_OFFSET_CONTROL
4537                            <1>       ;mov  eax, VIA_REG_CTRL_PAUSE + VIA_REG_CTRL_TERMINATE + VIA_REG_CTRL_RESET
4538 00014803 B848000000      <1>       mov   eax, VIA_REG_CTRL_PAUSE + VIA_REG_CTRL_TERMINATE ; 24/06/2017
4539 00014808 E88BFEFFFF      <1>       call  ctrl_io_w8
4540                            <1>
4541                            <1>       ;mov  edx, VIA_REG_OFFSET_CONTROL
4542                            <1>       ;call  ctrl_io_r8
4543                            <1>
4544                            <1>       ; wait for 50 ms
4545 0001480D B9A0000000      <1>       mov   ecx, 160 ; (200*0.25 ms) ; 29/05/2017
4546                            <1> _ch_rst_wait:
4547 00014812 E849FEFFFF      <1>       call  delay1_4ms
4548 00014817 49              <1>       dec   ecx
4549 00014818 75F8            <1>       jnz   short _ch_rst_wait
4550                            <1>
4551                            <1>       ; disable interrupts
4552 0001481A BA01000000      <1>       mov   edx, VIA_REG_OFFSET_CONTROL
4553 0001481F 31C0            <1>       xor   eax, eax
4554 00014821 E872FEFFFF      <1>       call  ctrl_io_w8
4555                            <1>
4556                            <1>       ; clear interrupts
4557 00014826 BA00000000      <1>       mov   edx, VIA_REG_OFFSET_STATUS
4558 0001482B B803000000      <1>       mov   eax, 3
4559 00014830 E863FEFFFF      <1>       call  ctrl_io_w8
4560                            <1>
4561                            <1>       ;mov  edx, VIA_REG_OFFSET_CURR_PTR
4562                            <1>       ;xor  eax, eax
4563                            <1>       ;call ctrl_io_w32
4564                            <1>
4565 00014835 C3              <1>       retn
4566                            <1>
4567                            <1> vt8233_stop: ; 22/04/2017
4568 00014836 C605[0C900100]00  <1>       mov   byte [audio_play_cmd], 0 ; stop !
4569                            <1> _tlp2:
4570                            <1>       ; 24/06/2017
4571                            <1>       ; finished with song, stop everything
4572                            <1>       ;mov  al, VIA_REG_CTRL_INT
4573                            <1>       ;or   al, VIA_REG_CTRL_TERMINATE
4574                            <1>       ;mov  dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
4575                            <1>       ;call ctrl_io_w8
4576                            <1>
4577                            <1>       ;call channel_reset
4578                            <1>       ;retn
4579                            <1>
4580 0001483D EBBF            <1>       jmp   short channel_reset
4581                            <1>
4582                            <1> set_vt8233_bdl: ; Set VT8237R Buffer Descriptor List
4583                            <1>       ; 22/07/2020 - TRDOS 386 v2.0.2
4584                            <1>       ; 28/05/2017
4585                            <1>       ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
4586                            <1>       ; 24/03/2017 - 'PLAYER.COM' ('via_wav.asm' - 29/11/2016)
4587                            <1>
4588                            <1>       ; eax = dma buffer address = [audio_DMA_buff]
4589                            <1>       ; ecx = dma buffer buffer size = [audio_dmabuff_size]
4590                            <1>
4591 0001483F D1E9              <1>       shr   ecx, 1 ; dma half buffer size
4592 00014841 89CE            <1>       mov   esi, ecx
4593                            <1>
4594 00014843 BF[14900100]      <1>       mov   edi, audio_bdl_buff ; get BDL address
4595 00014848 B910000000      <1>       mov   ecx, 32 / 2         ; make 32 entries in BDL
4596                            <1>
4597 0001484D EB05            <1>       jmp   short s_vt8233_bdl1
4598                            <1>
4599                            <1> s_vt8233_bdl0:
4600                            <1>       ; set buffer descriptor 0 to start of data file in memory
4601                            <1>
4602 0001484F A1[F88F0100]      <1>       mov   eax, [audio_dma_buff]     ; Physical address of DMA buffer
4603                            <1>
4604                            <1> s_vt8233_bdl1:
4605 00014854 AB              <1>       stosd                     ; store dmabuffer1 address
4606                            <1>
4607 00014855 89C2            <1>       mov   edx, eax
4608                            <1>
4609                            <1> ; VIA VT8235.PDF: (Page 110) (Erdogan Tan, 29/11/2016)
4610                            <1>       ;
4611                            <1>       ;     Audio SGD Table Format
4612                            <1>       ;     -------------------------------
4613                            <1>       ;     63   62    61-56    55-32  31-0
4614                            <1>       ;     --   --   --------  -----  ----
4615                            <1>       ;     EOL FLAG -reserved- Base   Base
4616                            <1>       ;                        Count  Address
4617                            <1>       ;                         [23:0] [31:0]
4618                            <1>       ;     EOL: End Of Link.
4619                            <1>       ;          1 indicates this block is the last of the link.
4620                            <1>       ;          If the channel "Interrupt on EOL" bit is set, then
4621                            <1>       ;          an interrupt is generated at the end of the transfer.
4622                            <1>       ;
4623                            <1>       ;     FLAG: Block Flag. If set, transfer pauses at the end of this
4624                            <1>       ;           block. If the channel "Interrupt on FLAG" bit is set,
4625                            <1>       ;           then an interrupt is generated at the end of this block.
4626                            <1>
4627 00014857 89F0            <1>       mov   eax, esi ; DMA half buffer size
4628 00014859 01C2            <1>       add   edx, eax
4629 0001485B 0D00000040      <1>       or    eax, FLAG
4630                            <1>       ;or   eax, EOL
4631 00014860 AB              <1>       stosd
4632                            <1>
```

```
                                <1> ; 2nd buffer:
                                <1>
00014861 89D0                   <1>       mov    eax, edx ; Physical address of the 2nd half of DMA buffer
00014863 AB                     <1>       stosd         ; store dmabuffer2 address
                                <1>
                                <1> ; set length to [audio_dmabuff_size]/2
                                <1> ; Set control (bits 31:16) to BUP, bits 15:0=number of samples
                                <1> ;
00014864 89F0                   <1>       mov    eax, esi ; DMA half buffer size
                                <1>       ; 22/07/2020
                                <1>       ;or    eax, EOL
00014866 0D00000040             <1>       or     eax, FLAG
0001486B AB                     <1>       stosd
                                <1>
0001486C E2E1                   <1>       loop    s_vt8233_bdl0
                                <1>
                                <1>       ; 22/07/2020
0001486E 814FFC00000080         <1>       or     dword [edi-4], EOL
                                <1>
00014875 C3                     <1>       retn
                                <1>
                                <1> vt8233_start_play:
                                <1>       ; 01/09/2020
                                <1>       ; 22/07/2020
                                <1>       ; start to play audio data via VT8233 audio controller
                                <1>       ; 13/06/2017
                                <1>       ; 10/06/2017
                                <1>       ; 24/04/2017
                                <1>       ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
                                <1>       ; 24/03/2017 - 'PLAYER.COM' ('via_wav.asm' - 29/11/2016)
                                <1>       ; write buffer descriptor list address
                                <1>
                                <1>       ; Extended Audio Status (2Ah)
00014876 B82A000000             <1>       mov    eax, CODEC_EXT_AUDIO_CTRL_REG ; 2Ah
0001487B E83CFEFFFF             <1>       call   codec_read
00014880 25FDFF0000             <1>       and    eax, 0FFFFh - 2     ; clear DRA (BIT1)
                                <1>       ;or    eax, 1              ; set VRA (BIT0)
                                <1>       ;or    eax, 5       ; VRA (BIT0) & S/PDIF (BIT2) ; 14/11/2016
00014885 0C05                   <1>       or     al, 5
                                <1>       ; 01/09/2020
                                <1>       ;or    eax, 3805h ; AD1980 (PRK, PRJ, PRI = 1 .. only front DAC)
                                <1>       ; 01/09/2020
                                <1>       ;mov   edx, CODEC_EXT_AUDIO_CTRL_REG
                                <1>       ;cmp   word [audio_freq], 0BB80h ; 48 kHz
                                <1>       ;jne   short set_extd_audio_status_1
                                <1>       ;and   al, 0FEh ; disable VRA bit (set sample rate to 48000 Hz)
                                <1>       ;jmp   short set_extd_audio_status_2
                                <1> ;set_extd_audio_status_1:
00014887 BA2A000000             <1>       mov    edx, CODEC_EXT_AUDIO_CTRL_REG
0001488C E857FEFFFF             <1>       call   codec_write
                                <1>       ;jc    short cconfig_error
                                <1>
                                <1> set_sample_rate:
                                <1>       ;movzx eax, word [audio_freq]
00014891 66A1[0A900100]         <1>       mov    ax, [audio_freq]
00014897 BA2C000000             <1>       mov    edx, CODEC_PCM_FRONT_DACRATE_REG ; 2Ch ; PCM Front DAC Rate
                                <1> ;set_extd_audio_status_2:
0001489C E847FEFFFF             <1>       call   codec_write
                                <1>
                                <1>       ; 01/09/2020
                                <1>       ; set AD1980 MCB register (Index 76h) to 0C00h
                                <1>       ; (CLDIS, HPSEL)
                                <1>       ;mov   ax, 0C00h
                                <1>       ;mov   edx, CODEC_MISC_CRTL_BITS_REG ; 76h
                                <1>       ;                   ; Miscellaneous Control Bit Register
                                <1>       ;call  codec_write
                                <1>       ;
                                <1>
000148A1 B8[14900100]           <1>       mov    eax, audio_bdl_buff
                                <1>
                                <1>       ; 12/11/2016 - Erdogan Tan
                                <1>       ; (Ref: KolibriOS, vt823x.asm, 'create_primary_buff')
000148A6 BA04000000             <1>       mov    edx, VIADEV_PLAYBACK + VIA_REG_OFFSET_TABLE_PTR
000148AB E8FAFDFFFF             <1>       call   ctrl_io_w32
                                <1>
                                <1>       ;call  codec_check_ready
                                <1>
000148B0 66BA0200               <1>       mov    dx, VIADEV_PLAYBACK + VIA_REG_OFS_PLAYBACK_VOLUME_L
                                <1>       ;mov   eax, 2 ; 31
000148B4 B01F                   <1>       mov    al, 31
000148B6 2A05[0E900100]         <1>       sub    al, [audio_master_volume_l]
000148BC E8D7FDFFFF             <1>       call   ctrl_io_w8
                                <1>
                                <1>       ;call  codec_check_ready
                                <1>
000148C1 66BA0300               <1>       mov    dx, VIADEV_PLAYBACK + VIA_REG_OFS_PLAYBACK_VOLUME_R
                                <1>       ;mov   ax, 2  ; 31
000148C5 B01F                   <1>       mov    al, 31
000148C7 2A05[0F900100]         <1>       sub    al, [audio_master_volume_r]
000148CD E8C6FDFFFF             <1>       call   ctrl_io_w8
                                <1>
                                <1>       ;call  codec_check_ready
                                <1> ;
                                <1> ;
                                <1> ; All set. Let's play some music.
                                <1> ;
                                <1> ;
                                <1>       ;mov   dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STOP_IDX
                                <1>       ;mov   ax, VIA8233_REG_TYPE_16BIT or VIA8233_REG_TYPE_STEREO or 0xfffff or 0xff000000
                                <1>       ;call  ctrl_io_w32
                                <1>
                                <1>       ;call  codec_check_ready
                                <1>
                                <1>       ; 08/12/2016
                                <1>       ; 07/10/2016
```

```
                                <1>       ;;mov    al, 1
                                <1>       ;mov   al, 31
                                <1>       ; 22/07/2020
000148D2 B0FF                   <1>       mov    al, 0FFh
000148D4 E813000000             <1>       call   set_VT8233_LastValidIndex
                                <1>
000148D9 C605[0C900100]01       <1>       mov    byte [audio_play_cmd], 1 ; play command (do not stop) !
                                <1>
                                <1>       ; 22/07/2020
                                <1>       ;mov   byte [audio_flag], 0  ; clear half buffer flag
                                <1>
                                <1> vt8233_play: ; continue to play
                                <1>       ; 22/04/2017
                                <1>       ;mov   al, VIA_REG_CTRL_INT
                                <1>       ;or    al, VIA_REG_CTRL_START
                                <1>       ;;mov  al, VIA_REG_CTRL_AUTOSTART + VIA_REG_CTRL_START
                                <1>       ; 22/07/2020
000148E0 B0A1                   <1>       mov    al, VIA_REG_CTRL_AUTOSTART + VIA_REG_CTRL_START + VIA_REG_CTRL_INT_FLAG
                                <1>
000148E2 66BA0100               <1>       mov    dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
000148E6 E8ADFDFFFF             <1>       call   ctrl_io_w8
                                <1>       ;call  codec_check_ready
                                <1>       ;retn
                                <1>       ;jmp   codec_check_ready
000148EB C3                     <1>       retn
                                <1>
                                <1> ;input AL = index # to stop on
                                <1> set_VT8233_LastValidIndex:
                                <1>       ; 23/07/2020
                                <1>       ; 10/06/2017
                                <1>       ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
                                <1>       ; 24/03/2017 - 'PLAYER.COM' ('via_wav.asm' - 29/11/2016)
                                <1>       ; 19/11/2016
                                <1>       ; 14/11/2016 - Erdogan Tan (Ref: VIA VT8235.PDF, Page 110)
                                <1>       ; 12/11/2016 - Erdogan Tan
                                <1>       ; (Ref: KolibriOS, vt823x.asm, 'create_primary_buff')
                                <1>       ;push  edx
                                <1>       ;push  ax
000148EC 50                     <1>       push   eax ; 23/07/2020
                                <1>       ;push  ecx
000148ED 0FB705[0A900100]       <1>       movzx  eax, word [audio_freq] ; Hertz
000148F4 BA00001000             <1>       mov    edx, 100000h ; 2^20 = 1048576
000148F9 F7E2                   <1>       mul    edx
000148FB B980BB0000             <1>       mov    ecx, 48000
00014900 F7F1                   <1>       div    ecx
                                <1>       ;and   eax, 0FFFFFh
                                <1>       ;pop   ecx
                                <1>       ;pop   dx
00014902 5A                     <1>       pop    edx ; 23/07/2020
00014903 C1E218                 <1>       shl    edx, 24  ; STOP Index Setting: Bit 24 to 31
00014906 09D0                   <1>       or     eax, edx
                                <1>       ; 19/11/2016
00014908 803D[08900100]10       <1>       cmp    byte [audio_bps], 16
0001490F 7505                   <1>       jne    short sLVI_1
00014911 0D00002000             <1>       or     eax, VIA8233_REG_TYPE_16BIT
                                <1> sLVI_1:
00014916 803D[09900100]02       <1>       cmp    byte [audio_stmo], 2
0001491D 7505                   <1>       jne    short sLVI_2
0001491F 0D00001000             <1>       or     eax, VIA8233_REG_TYPE_STEREO
                                <1> sLVI_2:
00014924 BA08000000             <1>       mov    edx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STOP_IDX
00014929 E87CFDFFFF             <1>       call   ctrl_io_w32
                                <1>       ;call  codec_check_ready
                                <1>       ;pop   edx
0001492E C3                     <1>       retn
                                <1>
                                <1> vt8233_pause: ; pause
                                <1>       ; 10/06/2017
                                <1>       ; 22/04/2017
                                <1>       ;mov   al, VIA_REG_CTRL_INT
                                <1>       ;or    al, VIA_REG_CTRL_PAUSE
                                <1>       ; 23/07/2020
0001492F B029                   <1>       mov    al, VIA_REG_CTRL_PAUSE+VIA_REG_CTRL_INT_FLAG+VIA_REG_CTRL_AUTOSTART
                                <1>
00014931 66BA0100               <1>       mov    dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
00014935 E85EFDFFFF             <1>       call   ctrl_io_w8
                                <1>       ;call  codec_check_ready
                                <1>       ;retn
                                <1>       ;jmp   codec_check_ready
0001493A C3                     <1>       retn
                                <1>
                                <1> vt8233_reset:
                                <1>       ; 22/04/2017
                                <1>       ; reset VT8237R (vt8233) Audio Controller
                                <1>       ;cmp   byte [audio_play_cmd], 1
                                <1>       ;jna   short vt8233_rst_0
0001493B C605[0C900100]00       <1>       mov    byte [audio_play_cmd], 0 ; stop !
                                <1> vt8233_rst_0:
00014942 E8B0FCFFFF             <1>       call   reset_codec
00014947 720A                   <1>       jc     short vt8233_rst_1 ; codec error !
                                <1>       ; eax = 1
00014949 E82EFDFFFF             <1>       call   codec_io_w16 ; w32
0001494E E8ABFEFFFF             <1>       call   channel_reset
                                <1> vt8233_rst_1:
00014953 C3                     <1>       retn
                                <1>
                                <1> vt8233_volume:
                                <1>       ; set VT8237R (vt8233) sound volume level
                                <1>       ; 24/04/2017
                                <1>       ; 22/04/2017
                                <1>       ; bl = component (0 = master/playback/lineout volume)
                                <1>       ; cl = left channel volume level (0 to 31)
                                <1>       ; ch = right channel volume level (0 to 31)
                                <1>
00014954 08DB                   <1>       or     bl, bl
```

```
4843 00014956 7520                <1>       jnz    short vt8233_vol_1 ; temporary !
4844 00014958 66B81F1F            <1>       mov    ax, 1F1Fh ; 31,31
4845 0001495C 38C1                <1>       cmp    cl, al
4846 0001495E 7718                <1>       ja     short vt8233_vol_1 ; temporary !
4847 00014960 38E5                <1>       cmp    ch, ah
4848 00014962 7714                <1>       ja     short vt8233_vol_1 ; temporary !
4849 00014964 66890D[0E900100]    <1>       mov    [audio_master_volume], cx
4850 0001496B 6629C8              <1>       sub    ax, cx
4851 0001496E BA02000000          <1>       mov    edx, CODEC_MASTER_VOL_REG ; 02h ; Line Out
4852 00014973 E870FDFFFF          <1>       call   codec_write
4853                              <1> vt8233_vol_1:
4854 00014978 C3                  <1>       retn
4855                              <1>
4856                              <1> ; CODE for SOUND BLASTER 16
4857                              <1>
4858                              <1> DetectSB:
4859                              <1>       ; 24/04/2017
4860                              <1>       ;pushad
4861                              <1> ScanPort:
4862 00014979 66BB1002            <1>       mov    bx, 210h     ; start scanning ports
4863                              <1>                           ; 210h, 220h, .. 260h
4864                              <1> ResetDSP:
4865 0001497D 6689DA              <1>       mov    dx, bx       ; try to reset the DSP.
4866 00014980 6683C206            <1>       add    dx, 06h
4867 00014984 B001                <1>       mov    al, 1
4868 00014986 EE                  <1>       out    dx, al
4869                              <1>
4870 00014987 EC                  <1>       in     al, dx
4871 00014988 EC                  <1>       in     al, dx
4872 00014989 EC                  <1>       in     al, dx
4873 0001498A EC                  <1>       in     al, dx
4874                              <1>
4875 0001498B 30C0                <1>       xor    al, al
4876 0001498D EE                  <1>       out    dx, al
4877                              <1>
4878 0001498E 6683C208            <1>       add    dx, 08h
4879 00014992 66B96400            <1>       mov    cx, 100
4880                              <1> WaitID:
4881 00014996 EC                  <1>       in     al, dx
4882 00014997 08C0                <1>       or     al, al
4883 00014999 7804                <1>       js     short GetID
4884 0001499B E2F9                <1>       loop   WaitID
4885 0001499D EB0F                <1>       jmp    short NextPort
4886                              <1> GetID:
4887 0001499F 6683EA04            <1>       sub    dx, 04h
4888 000149A3 EC                  <1>       in     al, dx
4889 000149A4 3CAA                <1>       cmp    al, 0AAh
4890 000149A6 7413                <1>       je     short Found
4891 000149A8 6683C204            <1>       add    dx, 04h
4892 000149AC E2E8                <1>       loop   WaitID
4893                              <1> NextPort:
4894 000149AE 6683C310            <1>       add    bx, 10h            ; if not response,
4895 000149B2 6681FB6002          <1>       cmp    bx, 260h           ; try the next port.
4896 000149B7 76C4                <1>       jbe    short ResetDSP
4897 000149B9 F9                  <1>       stc
4898 000149BA C3                  <1>       retn
4899                              <1> Found:
4900 000149BB 66891D[DE8F0100]    <1>       mov    [audio_io_base], bx ; SB Port Address Found!
4901                              <1> ScanIRQ:
4902                              <1> SetIrqs:
4903 000149C2 28C0                <1>       sub    al, al ; 0
4904 000149C4 A2[D48F0100]        <1>       mov    [IRQnum], al ; reset
4905 000149C9 A2[DB8F0100]        <1>       mov    [audio_intr], al ; reset
4906                              <1>
4907                              <1>       ; ah > 0 -> set IRQ vector
4908                              <1>       ; al = IRQ number
4909                              <1>       ;mov   ax, 103h ; IRQ 3
4910                              <1>       ;call  set_hardware_int_vector
4911                              <1>       ;mov   ax, 104h ; IRQ 4
4912                              <1>       ;call  set_hardware_int_vector
4913 000149CE 66B80501            <1>       mov    ax, 105h ; IRQ 5
4914 000149D2 E830E1FFFF          <1>       call   set_hardware_int_vector
4915 000149D7 66B80701            <1>       mov    ax, 107h ; IRQ 7
4916 000149DB E827E1FFFF          <1>       call   set_hardware_int_vector
4917                              <1>
4918 000149E0 668B15[DE8F0100]    <1>       mov    dx, [audio_io_base] ; tells to the SB to
4919 000149E7 6683C20C            <1>       add    dx, 0Ch             ; generate a IRQ!
4920                              <1> WaitSb:
4921 000149EB EC                  <1>       in     al, dx
4922 000149EC 08C0                <1>       or     al, al
4923 000149EE 78FB                <1>       js     short WaitSb
4924 000149F0 B0F2                <1>       mov    al, 0F2h
4925 000149F2 EE                  <1>       out    dx, al
4926                              <1>
4927 000149F3 31C9                <1>       xor    ecx, ecx     ; wait until IRQ level
4928                              <1> WaitIRQ:
4929 000149F5 A0[D48F0100]        <1>       mov    al, [IRQnum]
4930 000149FA 3C00                <1>       cmp    al, 0 ; is changed or timeout.
4931 000149FC 7706                <1>       ja     short IrqOk
4932 000149FE 6649                <1>       dec    cx
4933 00014A00 75F3                <1>       jnz    short WaitIRQ
4934 00014A02 EB15                <1>       jmp    short RestoreIrqs
4935                              <1> IrqOk:
4936 00014A04 A2[DB8F0100]        <1>       mov    [audio_intr], al ; set
4937 00014A09 668B15[DE8F0100]    <1>       mov    dx, [audio_io_base]
4938 00014A10 6683C20E            <1>       add    dx, 0Eh
4939 00014A14 EC                  <1>       in     al, dx ; SB acknowledge.
4940 00014A15 B020                <1>       mov    al, 20h
4941 00014A17 E620                <1>       out    20h, al      ; Hardware acknowledge.
4942                              <1>
4943                              <1> RestoreIrqs:
4944                              <1>       ; ah = 0 -> reset IRQ vector
4945                              <1>       ; al = IRQ number
4946                              <1>       ;mov   ax, 3 ; IRQ 3
4947                              <1>       ;call  set_hardware_int_vector
```

```
4948                                    <1>       ;mov   ax, 4 ; IRQ 4
4949                                    <1>       ;call  set_hardware_int_vector
4950 00014A19 66B80500                 <1>       mov    ax, 5 ; IRQ 5
4951 00014A1D E8E5E0FFFF               <1>       call   set_hardware_int_vector
4952 00014A22 66B80700                 <1>       mov    ax, 7 ; IRQ 7
4953 00014A26 E8DCE0FFFF               <1>       call   set_hardware_int_vector
4954                                    <1>
4955 00014A2B 31D2                     <1>       xor    edx, edx
4956 00014A2D 8915[E08F0100]           <1>       mov    [audio_dev_id], edx ; 0
4957 00014A33 8915[E48F0100]           <1>       mov    [audio_vendor], edx ; 0
4958 00014A39 8915[E88F0100]           <1>       mov    [audio_stats_cmd], edx ; 0
4959                                    <1>
4960                                    <1>       ;popad
4961                                    <1>
4962 00014A3F 803D[DB8F0100]01         <1>       cmp    byte [audio_intr], 1 ; IRQ level was changed?
4963                                    <1>
4964 00014A46 C3                       <1>       retn
4965                                    <1>
4966                                    <1> %macro      SbOut  1
4967                                    <1> %%Wait:
4968                                    <1>       in     al, dx
4969                                    <1>       or     al, al
4970                                    <1>       js     short %%Wait
4971                                    <1>       mov    al, %1
4972                                    <1>       out    dx, al
4973                                    <1> %endmacro
4974                                    <1>
4975                                    <1> SbInit_play:
4976                                    <1>       ; 22/10/2017
4977                                    <1>       ; 20/10/2017
4978                                    <1>       ; 06/10/2017
4979                                    <1>       ; 13/07/2017, 09/08/2017
4980                                    <1>       ; 24/04/2017, 15/05/2017, 24/06/2017
4981                                    <1>       ;pushad
4982                                    <1> SetBuffer:
4983                                    <1>       ;mov   byte [DmaFlag], 0
4984                                    <1>
4985 00014A47 8B1D[F88F0100]           <1>       mov    ebx, [audio_dma_buff] ; physical addr of DMA buff
4986 00014A4D 89DF                     <1>       mov    edi, ebx
4987 00014A4F 8B0D[FC8F0100]           <1>       mov    ecx, [audio_dmabuff_size]
4988                                    <1>
4989 00014A55 803D[08900100]10         <1>       cmp    byte [audio_bps], 16
4990 00014A5C 7531                     <1>       jne    short sbInit_0 ; set 8 bit DMA buffer
4991                                    <1>
4992                                    <1>       ; 09/08/2017
4993                                    <1>       ; convert byte count to word count
4994 00014A5E D1E9                     <1>       shr    ecx, 1
4995 00014A60 49                       <1>       dec    ecx ; word count - 1
4996                                    <1>       ; convert byte offset to word offset
4997 00014A61 D1EB                     <1>       shr    ebx, 1
4998                                    <1>
4999                                    <1>       ; 16 bit DMA buffer setting (DMA channel 5)
5000 00014A63 B005                     <1>       mov    al, 05h  ; set mask bit for channel 5  (4+1)
5001 00014A65 E6D4                     <1>       out    0D4h, al
5002                                    <1>
5003 00014A67 30C0                     <1>       xor    al, al   ; stops all DMA processes on selected channel
5004 00014A69 E6D8                     <1>       out    0D8h, al ; clear selected channel register
5005                                    <1>
5006 00014A6B 88D8                     <1>       mov    al, bl  ; byte 0 of DMA buffer offset in words (physical)
5007 00014A6D E6C4                     <1>       out    0C4h, al ; DMA channel 5 port number
5008                                    <1>
5009 00014A6F 88F8                     <1>       mov    al, bh   ; byte 1 of DMA buffer offset in words (physical)
5010 00014A71 E6C4                     <1>       out    0C4h, al
5011                                    <1>
5012                                    <1>       ; 09/08/2017
5013 00014A73 C1EB0F                   <1>       shr    ebx, 15       ; complete 16 bit shift
5014 00014A76 80E3FE                   <1>       and    bl, 0FEh ; clear bit 0 (not necessary, it will be ignored)
5015                                    <1>
5016 00014A79 88D8                     <1>       mov    al, bl   ; byte 2 of DMA buffer address (physical)
5017 00014A7B E68B                     <1>       out    8Bh, al  ; page register port addr for channel 5 ; 13/07/2017
5018                                    <1>
5019 00014A7D 88C8                     <1>       mov    al, cl   ; low byte of DMA count - 1
5020 00014A7F E6C6                     <1>       out    0C6h, al ; count register port addr for channel 1
5021                                    <1>
5022 00014A81 88E8                     <1>       mov    al, ch   ; high byte of DMA count - 1
5023 00014A83 E6C6                     <1>       out    0C6h, al
5024                                    <1>
5025                                    <1>       ; channel 5, read, autoinitialized, single mode
5026                                    <1>       ;mov   al, 49h
5027 00014A85 B059                     <1>       mov    al, 59h  ; 06/10/2017
5028 00014A87 E6D6                     <1>       out    0D6h, al ; DMA mode register port address
5029                                    <1>
5030 00014A89 B001                     <1>       mov    al, 01h  ; clear mask bit for channel 1
5031 00014A8B E6D4                     <1>       out    0D4h, al ; DMA mask register port address
5032                                    <1>
5033 00014A8D EB28                     <1>       jmp    short ClearBuffer
5034                                    <1>
5035                                    <1> sbInit_0:
5036 00014A8F 49                       <1>       dec    ecx ; 09/08/2017
5037                                    <1>
5038                                    <1>       ; 8 bit DMA buffer setting (DMA channel 1)
5039 00014A90 B005                     <1>       mov    al, 05h ; set mask bit for channel 1  (4+1)
5040 00014A92 E60A                     <1>       out    0Ah, al ; DMA mask register
5041                                    <1>
5042 00014A94 30C0                     <1>       xor    al, al  ; stops all DMA processes on selected channel
5043 00014A96 E60C                     <1>       out    0Ch, al ; clear selected channel register
5044                                    <1>
5045 00014A98 88D8                     <1>       mov    al, bl ; byte 0 of DMA buffer address (physical)
5046 00014A9A E602                     <1>       out    02h, al ; DMA channel 1 port number
5047                                    <1>
5048 00014A9C 88F8                     <1>       mov    al, bh  ; byte 1 of DMA buffer address (physical)
5049 00014A9E E602                     <1>       out    02h, al
5050                                    <1>
5051 00014AA0 C1EB10                   <1>       shr    ebx, 16
5052                                    <1>
```

```
5053 00014AA3 88D8                <1>       mov    al, bl  ; byte 2 of DMA buffer address (physical)
5054 00014AA5 E683                <1>       out    83h, al ; page register port addr for channel 1
5055                              <1>
5056 00014AA7 88C8                <1>       mov    al, cl  ; low byte of DMA count - 1
5057 00014AA9 E603                <1>       out    03h, al ; count register port addr for channel 1
5058                              <1>
5059 00014AAB 88E8                <1>       mov    al, ch  ; high byte of DMA count - 1
5060 00014AAD E603                <1>       out    03h, al
5061                              <1>
5062                              <1>       ; channel 1, read, autoinitialized, single mode
5063                              <1>       ;mov   al, 49h
5064 00014AAF B059                <1>       mov    al, 59h ; 06/10/2017
5065 00014AB1 E60B                <1>       out    0Bh, al ; DMA mode register port address
5066                              <1>
5067 00014AB3 B001                <1>       mov    al, 01h ; clear mask bit for channel 1
5068 00014AB5 E60A                <1>       out    0Ah, al ; DMA mask register port address
5069                              <1>
5070                              <1> ClearBuffer:
5071                              <1>       ;;mov  edi, [audio_dma_buff]
5072                              <1>       ;;mov  ecx, [audio_dmabuff_size]
5073                              <1>       ;inc   ecx
5074                              <1>       ;mov   al, 80h
5075                              <1>       ;;cld
5076                              <1>       ;rep   stosb
5077                              <1> SetIrq:
5078                              <1>       ;mov   ebx, SbIrqhandler
5079                              <1>       ;mov   al, [audio_intr] ; IRQ number
5080                              <1>       ;call  set_dev_IRQ_service
5081                              <1>       ;; SETUP (audio) INTERRUPT CALLBACK SERVICE
5082                              <1>       ;mov   bl, [audio_intr] ; IRQ number
5083                              <1>       ;mov   bh, [audio_cb_mode]
5084                              <1>       ;inc   bh  ; 1 = Signal Response Byte method (fixed value)
5085                              <1>       ;          ; 2 = Callback service method
5086                              <1>       ;          ; 3 = Auto Increment S.R.B. method
5087                              <1>       ;mov   cl, [audio_srb]
5088                              <1>       ;mov   edx, [audio_cb_addr]
5089                              <1>       ;mov   al, [audio_user]
5090                              <1>       ;call  set_irq_callback_service
5091                              <1> ResetDsp:
5092 00014AB7 668B15[DE8F0100]    <1>       mov    dx, [audio_io_base]
5093 00014ABE 6683C206            <1>       add    dx, 06h
5094 00014AC2 B001                <1>       mov    al, 1
5095 00014AC4 EE                  <1>       out    dx, al
5096                              <1>
5097 00014AC5 EC                  <1>       in     al, dx
5098 00014AC6 EC                  <1>       in     al, dx
5099 00014AC7 EC                  <1>       in     al, dx
5100 00014AC8 EC                  <1>       in     al, dx
5101                              <1>
5102 00014AC9 30C0                <1>       xor    al, al
5103 00014ACB EE                  <1>       out    dx, al
5104                              <1>
5105 00014ACC 66B96400            <1>       mov    cx, 100
5106 00014AD0 28E4                <1>       sub    ah, ah ; 0
5107                              <1> WaitId:
5108 00014AD2 668B15[DE8F0100]    <1>       mov    dx, [audio_io_base]
5109 00014AD9 6683C20E            <1>       add    dx, 0Eh
5110 00014ADD EC                  <1>       in     al, dx
5111 00014ADE 08C0                <1>       or     al, al
5112 00014AE0 7807                <1>       js     short sb_GetId
5113 00014AE2 E2EE                <1>       loop   WaitId
5114 00014AE4 E9B4000000          <1>       jmp    sb_Exit
5115                              <1> sb_GetId:
5116 00014AE9 668B15[DE8F0100]    <1>       mov    dx, [audio_io_base]
5117 00014AF0 6683C20A            <1>       add    dx, 0Ah
5118 00014AF4 EC                  <1>       in     al, dx
5119 00014AF5 3CAA                <1>       cmp    al, 0AAh
5120 00014AF7 7407                <1>       je     short SbOk
5121 00014AF9 E2D7                <1>       loop   WaitId
5122 00014AFB E99D000000          <1>       jmp    sb_Exit
5123                              <1> SbOk:
5124 00014B00 668B15[DE8F0100]    <1>       mov    dx, [audio_io_base]
5125 00014B07 6683C20C            <1>       add    dx, 0Ch
5126                              <1>       SbOut  0D1h ; Turn on speaker
4967                              <2> %%Wait:
4968 00014B0B EC                  <2>       in al, dx
4969 00014B0C 08C0                <2>       or al, al
4970 00014B0E 78FB                <2>       js short %%Wait
4971 00014B10 B0D1                <2>       mov al, %1
4972 00014B12 EE                  <2>       out dx, al
5127                              <1>       SbOut   41h ; 8 bit or 16 bit transfer
4967                              <2> %%Wait:
4968 00014B13 EC                  <2>       in al, dx
4969 00014B14 08C0                <2>       or al, al
4970 00014B16 78FB                <2>       js short %%Wait
4971 00014B18 B041                <2>       mov al, %1
4972 00014B1A EE                  <2>       out dx, al
5128 00014B1B 668B1D[0A900100]    <1>       mov    bx, [audio_freq] ; sampling rate (Hz)
5129                              <1>       SbOut  bh ; sampling rate high byte
4967                              <2> %%Wait:
4968 00014B22 EC                  <2>       in al, dx
4969 00014B23 08C0                <2>       or al, al
4970 00014B25 78FB                <2>       js short %%Wait
4971 00014B27 88F8                <2>       mov al, %1
4972 00014B29 EE                  <2>       out dx, al
5130                              <1>       SbOut  bl ; sampling rate low byte
4967                              <2> %%Wait:
4968 00014B2A EC                  <2>       in al, dx
4969 00014B2B 08C0                <2>       or al, al
4970 00014B2D 78FB                <2>       js short %%Wait
4971 00014B2F 88D8                <2>       mov al, %1
4972 00014B31 EE                  <2>       out dx, al
5131                              <1>
5132                              <1>       ; 22/05/2017
5133 00014B32 E8C0000000          <1>       call   sb16_volume_initial ; 15/05/2017
```

```
                            <1>       ; 20/05/2017
                            <1>       ;call  sb16_volume
                            <1>
                            <1> StartDma:
                            <1>       ; autoinitialized mode
00014B37 803D[08900100]10   <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
00014B3E 7411               <1>       je     short sb_play_1
                            <1>       ; 8 bit samples
00014B40 66BBC600           <1>       mov    bx, 0C6h ; 8 bit output (0C6h)
00014B44 803D[09900100]02   <1>       cmp    byte [audio_stmo], 2 ; 1 = mono, 2 = stereo
00014B4B 7214               <1>       jb     short sb_play_2
00014B4D B720               <1>       mov    bh, 20h      ; 8 bit stereo (20h)
00014B4F EB10               <1>       jmp    short sb_play_2
                            <1> sb_play_1:
                            <1>       ; 16 bit samples
00014B51 66BBB610           <1>       mov    bx, 10B6h ; 16 bit output (0B6h)
00014B55 803D[09900100]02   <1>       cmp    byte [audio_stmo], 2 ; 1 = mono, 2 = stereo
00014B5C 7203               <1>       jb     short sb_play_2
00014B5E 80C720             <1>       add    bh, 20h      ; 16 bit stereo (30h)
                            <1> sb_play_2:
                            <1>       ; PCM output (8/16 bit mono autoinitialized transfer)
                            <1>       SbOut  bl ; bCommand
                            <2> %%Wait:
00014B61 EC                 <2>       in al, dx
00014B62 08C0               <2>       or al, al
00014B64 78FB               <2>       js short %%Wait
00014B66 88D8               <2>       mov al, %1
00014B68 EE                 <2>       out dx, al
                            <1>       SbOut  bh ; bMode
                            <2> %%Wait:
00014B69 EC                 <2>       in al, dx
00014B6A 08C0               <2>       or al, al
00014B6C 78FB               <2>       js short %%Wait
00014B6E 88F8               <2>       mov al, %1
00014B70 EE                 <2>       out dx, al
00014B71 8B1D[FC8F0100]     <1>       mov    ebx, [audio_dmabuff_size]  ; 15/05/2017
00014B77 D1EB               <1>       shr    ebx, 1 ; half buffer size
                            <1>       ; 20/10/2017
00014B79 803D[08900100]10   <1>       cmp    byte [audio_bps], 16 ; 16 bit DMA
00014B80 7502               <1>       jne    short sb_play_3
00014B82 D1EB               <1>       shr    ebx, 1 ; byte count to word count
                            <1> sb_play_3:
00014B84 664B               <1>       dec    bx  ; wBlkSize is one less than the actual size
                            <1>       SbOut  bl
                            <2> %%Wait:
00014B86 EC                 <2>       in al, dx
00014B87 08C0               <2>       or al, al
00014B89 78FB               <2>       js short %%Wait
00014B8B 88D8               <2>       mov al, %1
00014B8D EE                 <2>       out dx, al
                            <1>       SbOut  bh
                            <2> %%Wait:
00014B8E EC                 <2>       in al, dx
00014B8F 08C0               <2>       or al, al
00014B91 78FB               <2>       js short %%Wait
00014B93 88F8               <2>       mov al, %1
00014B95 EE                 <2>       out dx, al
                            <1>
00014B96 C605[0C900100]01   <1>       mov    byte [audio_play_cmd], 1 ; playing !
                            <1>
                            <1>       ;; Set Voice and master volumes
                            <1>       ;mov   dx, [audio_io_base]
                            <1>       ;add   dl, 4 ; Mixer chip Register Address Port
                            <1>       ;SbOut 30h   ; select Master Volume Register (L)
                            <1>       ;inc   dl    ; Mixer chip Register Data Port
                            <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
                            <1>       ;dec   dl
                            <1>       ;SbOut 31h   ; select Master Volume Register (R)
                            <1>       ;inc   dl
                            <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
                            <1>       ;dec   dl
                            <1>       ;SbOut 32h   ; select Voice Volume Register (L)
                            <1>       ;inc   dl
                            <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
                            <1>       ;dec   dl
                            <1>       ;SbOut 33h   ; select Voice Volume Register (R)
                            <1>       ;inc   dl
                            <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
                            <1>       ;;
                            <1>       ;dec   dl
                            <1>       ;SbOut 44h   ; select Treble Register (L)
                            <1>       ;inc   dl
                            <1>       ;SbOut 0F0h  ; Max. Treble value is 15 (15*16)
                            <1>       ;dec   dl
                            <1>       ;SbOut 45h   ; select Treble Register (R)
                            <1>       ;inc   dl
                            <1>       ;SbOut 0F0h  ; Max. Treble value is 15 (15*16)
                            <1>       ;dec   dl
                            <1>       ;SbOut 46h   ; select Bass Register (L)
                            <1>       ;inc   dl
                            <1>       ;SbOut 0F0h  ; Max. Bass value is 15 (15*16)
                            <1>       ;dec   dl
                            <1>       ;SbOut 47h   ; select Bass Register (R)
                            <1>       ;inc   dl
                            <1>       ;SbOut 0F0h  ; Max. Bass value is 15 (15*16)
                            <1>
                            <1> sb_Exit:
                            <1>       ;popad
00014B9D C3                 <1>       retn
                            <1>
                            <1> sb16_int_handler:
                            <1>       ; Interrupt Handler for Sound Blaster 16 Audio Card
                            <1>       ; Note: called by 'dev_IRQ_service'
                            <1>       ; 20/10/2017
                            <1>       ; 12/10/2017
```

```
                                <1>       ; 10/10/2017
                                <1>       ; 12/05/2017, 09/10/2017
                                <1>       ; 24/04/2017 (TRDOS 386 kernel, 'audio.s')
                                <1>       ; 10/03/2017 - 'PLAYWAV.PRG' ('playwav.s')
                                <1>
                                <1>       ;push  eax ; * must be saved !
                                <1>       ;push  ebx ; * must be saved !
                                <1>       ;push  ecx
                                <1>       ;push  edx
                                <1>       ;push  esi
                                <1>       ;push  edi
                                <1>
00014B9E 668B15[DE8F0100]       <1>       mov    dx, [audio_io_base]
                                <1>       ; 20/10/2017
00014BA5 80C20F                 <1>       add    dl, 0Fh ; 2xFh (DSP 16 bit intr ack)
00014BA8 803D[08900100]10       <1>       cmp    byte [audio_bps], 16
00014BAF 7402                   <1>       je     short sb_irq_16bit_ack
                                <1> sb_irq_8bit_ack:
00014BB1 FECA                   <1>       dec    dl  ; 2xEh (DSP 8 bit intr ack)
                                <1> sb_irq_16bit_ack:
00014BB3 EC                     <1>       in     al, dx
                                <1>
                                <1>       ;cmp   byte [audio_busy], 0
                                <1>       ;ja    short sb_irq_h3
                                <1>
                                <1>       ;mov   byte [audio_busy], 1
                                <1>
00014BB4 803D[0C900100]01       <1>       cmp    byte [audio_play_cmd], 1
00014BBB 7307                   <1>       jnb    short sb_irq_h1
                                <1> sb_irq_h0:
00014BBD E8A9000000             <1>       call   sb16_stop
00014BC2 EB2B                   <1>       jmp    short sb_irq_h3
                                <1> sb_irq_h1:
                                <1>       ;call  sb16_tuneloop
                                <1>       ; 09/10/2017
                                <1> sb16_tuneloop:
00014BC4 8B3D[F88F0100]         <1>       mov    edi, [audio_dma_buff]
00014BCA 8B0D[FC8F0100]         <1>       mov    ecx, [audio_dmabuff_size]
00014BD0 D1E9                   <1>       shr    ecx, 1 ; dma buff size / 2 = half buffer size
                                <1>
                                <1>       ; 22/05/2017
00014BD2 F605[00900100]01       <1>       test   byte [audio_flag], 1  ; Current flag value
00014BD9 7402                   <1>       jz     short sb_tlp1 ; EOL (Half Buffer 1 must be filled)
                                <1>       ; FLAG (Half Buffer 2 must be filled)
00014BDB 01CF                   <1>       add    edi, ecx
                                <1>       ; 15/05/2017
                                <1> sb_tlp1:
00014BDD 8B35[F08F0100]         <1>       mov    esi, [audio_p_buffer] ; phy addr of audio buff
                                <1>       ;rep   movsb
00014BE3 C1E902                 <1>       shr    ecx, 2 ; half buff size / 4
00014BE6 F3A5                   <1>       rep    movsd
                                <1>       ;retn
                                <1>
                                <1>       ; 10/10/2017
                                <1>       ; switch flag value
00014BE8 8035[00900100]01       <1>       xor    byte [audio_flag], 1
                                <1>
                                <1>       ; 12/10/2017
                                <1>       ; [audio_flag] = 0 : Playing dma half buffer 2 (odd intr count)
                                <1>                          ; Next buffer (to update) is dma half buff 1
                                <1>       ;              = 1 : Playing dma half buffer 1 (even intr count)
                                <1>                            ; Next buffer (to update) is dma half buff 2
                                <1>
                                <1> sb_irq_h3:
                                <1>       ;mov   byte [audio_busy], 0
                                <1>
                                <1>       ;pop   edi
                                <1>       ;pop   esi
                                <1>       ;pop   edx
                                <1>       ;pop   ecx
                                <1>       ;pop   ebx ; * must be restored !
                                <1>       ;pop   eax ; * must be restored !
                                <1>
00014BEF C3                     <1>       retn
                                <1>
                                <1> sb16_volume:
                                <1>       ; 22/10/2017
                                <1>       ; mov [audio_master_volume_l], cl
                                <1>       ; mov [audio_master_volume_h], ch
00014BF0 66890D[0E900100]       <1>       mov    [audio_master_volume], cx
                                <1> sb16_volume_initial:
00014BF7 6652                   <1>       push   dx ; DX (port address) must be saved
00014BF9 668B15[DE8F0100]       <1>       mov    dx, [audio_io_base]
00014C00 6683C204               <1>       add    dx, 4 ; Mixer chip address port
00014C04 B022                   <1>       mov    al, 22h ; master volume
00014C06 EE                     <1>       out    dx, al
00014C07 6642                   <1>       inc    dx
00014C09 8A25[0E900100]         <1>       mov    ah, [audio_master_volume_l]
00014C0F C0EC02                 <1>       shr    ah, 2 ; 32 -> 8 level
00014C12 C0E405                 <1>       shl    ah, 5 ; bit 5 to 7
00014C15 A0[0F900100]           <1>       mov    al, [audio_master_volume_r]
00014C1A C0E802                 <1>       shr    al, 2 ; 32 -> 8 level
                                <1>       ;and   al, 0Fh
00014C1D D0E0                   <1>       shl    al, 1 ; bit 1 to 3
00014C1F 08E0                   <1>       or     al, ah
00014C21 EE                     <1>       out    dx, al
00014C22 665A                   <1>       pop    dx ; DX (port address) must be restored
00014C24 C3                     <1>       retn
                                <1>
                                <1> sb16_pause:
00014C25 668B15[DE8F0100]       <1>       mov    dx, [audio_io_base]
00014C2C 6683C20C               <1>       add    dx, 0Ch ; Command & Data Port
00014C30 803D[08900100]10       <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
00014C37 7404                   <1>       je     short sb_pause_1
                                <1>       ; 8 bit samples
```

```
5320 00014C39 B3D0              <1>       mov    bl, 0D0h ; 8 bit DMA mode
5321 00014C3B EB02              <1>       jmp    short sb_pause_2
5322                            <1> sb_pause_1:
5323                            <1>       ; 16 bit samples
5324 00014C3D B3D5              <1>       mov    bl, 0D5h ; 16 bit DMA mode
5325                            <1> sb_pause_2:
5326                            <1>       SbOut   bl ; bCommand
4967                            <2> %%Wait:
4968 00014C3F EC                <2>       in al, dx
4969 00014C40 08C0              <2>       or al, al
4970 00014C42 78FB              <2>       js short %%Wait
4971 00014C44 88D8              <2>       mov al, %1
4972 00014C46 EE                <2>       out dx, al
5327                            <1> sb_pause_3:
5328 00014C47 C3                <1>       retn
5329                            <1>
5330                            <1> sb16_continue:
5331 00014C48 668B15[DE8F0100]  <1>       mov    dx, [audio_io_base]
5332 00014C4F 6683C20C          <1>       add    dx, 0Ch ; Command & Data Port
5333 00014C53 803D[08900100]10  <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
5334 00014C5A 7404              <1>       je     short sb_cont_1
5335                            <1>       ; 8 bit samples
5336 00014C5C B3D4              <1>       mov    bl, 0D4h ; 8 bit DMA mode
5337 00014C5E EB02              <1>       jmp    short sb_cont_2
5338                            <1> sb_cont_1:
5339                            <1>       ; 16 bit samples
5340 00014C60 B3D6              <1>       mov    bl, 0D6h ; 16 bit DMA mode
5341                            <1> sb_cont_2:
5342                            <1>       SbOut  bl ; bCommand
4967                            <2> %%Wait:
4968 00014C62 EC                <2>       in al, dx
4969 00014C63 08C0              <2>       or al, al
4970 00014C65 78FB              <2>       js short %%Wait
4971 00014C67 88D8              <2>       mov al, %1
4972 00014C69 EE                <2>       out dx, al
5343                            <1> sb_cont_3:
5344 00014C6A C3                <1>       retn
5345                            <1>
5346                            <1> sb16_stop:
5347                            <1>       ; 24/04/2017
5348 00014C6B 803D[0C900100]00  <1>       cmp    byte [audio_play_cmd], 0
5349 00014C72 7648              <1>       jna    short sb16_stop_4
5350                            <1>
5351                            <1>       ; 22/05/2017
5352 00014C74 668B15[DE8F0100]  <1>       mov    dx, [audio_io_base]
5353 00014C7B 6683C20C          <1>       add    dx, 0Ch
5354                            <1>
5355 00014C7F B3D9              <1>       mov    bl, 0D9h ; exit auto-initialize 16 bit transfer
5356                            <1>       ; stop  autoinitialized DMA transfer mode
5357 00014C81 803D[08900100]10  <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
5358 00014C88 7402              <1>       je     short sb16_stop_1
5359                            <1>       ;mov   bl, 0DAh ; exit auto-initialize 8 bit transfer
5360 00014C8A FEC3              <1>       inc    bl
5361                            <1> sb16_stop_1:
5362                            <1>       SbOut  bl ; exit auto-initialize transfer command
4967                            <2> %%Wait:
4968 00014C8C EC                <2>       in al, dx
4969 00014C8D 08C0              <2>       or al, al
4970 00014C8F 78FB              <2>       js short %%Wait
4971 00014C91 88D8              <2>       mov al, %1
4972 00014C93 EE                <2>       out dx, al
5363                            <1>
5364 00014C94 30C0              <1>       xor     al, al ; stops all DMA processes on selected channel
5365                            <1>
5366 00014C96 803D[08900100]10  <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
5367 00014C9D 7404              <1>       je     short sb16_stop_2
5368 00014C9F E60C              <1>       out    0Ch, al ; clear selected channel register
5369 00014CA1 EB02              <1>       jmp    short sb16_stop_3
5370                            <1>
5371                            <1> sb16_stop_2:
5372 00014CA3 E6D8              <1>       out    0D8h, al ; clear selected channel register
5373                            <1>
5374                            <1> sb16_stop_3:
5375 00014CA5 C605[0C900100]00  <1>       mov    byte [audio_play_cmd], 0 ; stop !
5376                            <1> SbDone:
5377                            <1>       ;mov   dx, [audio_io_base]
5378                            <1>       ;add   dx, 0Ch
5379                            <1>       SbOut  0D0h
4967                            <2> %%Wait:
4968 00014CAC EC                <2>       in al, dx
4969 00014CAD 08C0              <2>       or al, al
4970 00014CAF 78FB              <2>       js short %%Wait
4971 00014CB1 B0D0              <2>       mov al, %1
4972 00014CB3 EE                <2>       out dx, al
5380                            <1>       SbOut  0D3h
4967                            <2> %%Wait:
4968 00014CB4 EC                <2>       in al, dx
4969 00014CB5 08C0              <2>       or al, al
4970 00014CB7 78FB              <2>       js short %%Wait
4971 00014CB9 B0D3              <2>       mov al, %1
4972 00014CBB EE                <2>       out dx, al
5381                            <1> sb16_stop_4:
5382 00014CBC C3                <1>       retn
5383                            <1>
5384                            <1> sb16_reset:
5385                            <1>       ; 24/04/2017
5386 00014CBD 668B15[DE8F0100]  <1>       mov     dx, [audio_io_base] ; try to reset the DSP.
5387 00014CC4 6683C206          <1>       add     dx, 06h
5388 00014CC8 B001              <1>       mov    al, 1
5389 00014CCA EE                <1>       out    dx, al
5390                            <1>
5391 00014CCB EC                <1>       in     al, dx
5392 00014CCC EC                <1>       in     al, dx
5393 00014CCD EC                <1>       in     al, dx
5394 00014CCE EC                <1>       in     al, dx
```

```
                                <1>
00014CCF 30C0                   <1>       xor    al, al
00014CD1 EE                     <1>       out    dx, al
                                <1>
00014CD2 6683C208               <1>       add    dx, 08h
00014CD6 66B96400               <1>       mov    cx, 100
                                <1> sbrstWaitID:
00014CDA EC                     <1>       in     al, dx
00014CDB 08C0                   <1>       or     al, al
00014CDD 7804                   <1>       js     short sbrstGetID
00014CDF E2F9                   <1>       loop   sbrstWaitID
00014CE1 F9                     <1>       stc
00014CE2 C3                     <1>       retn
                                <1> sbrstGetID:
00014CE3 6683EA04               <1>       sub    dx, 04h
00014CE7 EC                     <1>       in     al, dx
00014CE8 3CAA                   <1>       cmp    al, 0AAh
00014CEA 7406                   <1>       je     short sb_rst_retn
00014CEC 6683C204               <1>       add    dx, 04h
00014CF0 E2E8                   <1>       loop   sbrstWaitID
                                <1> sb_rst_retn:
00014CF2 C3                     <1>       retn
                                <1>
                                <1> ac97_codec_config:
                                <1>       ; 10/06/2017
                                <1>       ; 05/06/2017
                                <1>       ; 29/05/2017
                                <1>       ; 28/05/2017 (TRDOS 386, 'audio.s')
                                <1>       ; 07/11/2016 (Erdogan Tan)
                                <1>       ; Derived from 'codecConfig' procedure in 'CODEC.ASM'
                                <1>       ; .wav player for DOS by Jeff Leyda (02/09/2002)
                                <1>
                                <1>       ;; 'PLAYER.ASM'
                                <1>       ;; get ICH base address regs for mixer and bus master
                                <1>
                                <1> init_ac97_controller: ; 10/06/2017
00014CF3 A1[E08F0100]           <1>       mov    eax, [audio_dev_id]
                                <1>       ;mov   al, NAMBAR_REG
                                <1>       ;;call  pciRegRead16            ; read PCI registers 10-11
                                <1>       ;call  pciRegRead32
                                <1>       ;and   dx, IO_ADDR_MASK         ; mask off BIT0
                                <1>       ;;and  edx, IO_ADDR_MASK
                                <1>
                                <1>       ;mov   [NAMBAR], dx             ; save audio mixer base addr
                                <1>
                                <1>       ;mov   al, NABMBAR_REG
                                <1>       ;;call pciRegRead16
                                <1>       ;call  pciRegRead32
                                <1>       ;and   dx, 0FFC0h ; IO_ADDR_MASK
                                <1>       ;;and  edx, 0FFC0h
                                <1>
                                <1>       ;mov   [NABMBAR], dx            ; save bus master base addr
                                <1>
                                <1>       ;mov   eax, [audio_dev_id]
00014CF8 B004                   <1>       mov    al, PCI_CMD_REG
                                <1>       ;call  pciRegRead8              ; read PCI command register
00014CFA E84AF8FFFF             <1>       call   pciRegRead16
00014CFF 80CA05                 <1>       or     dl, IO_ENA+BM_ENA        ; enable IO and bus master
                                <1>       ;call  pciRegWrite8
00014D02 E8ADF8FFFF             <1>       call   pciRegWrite16
                                <1>
                                <1>       ; 'CODEC.ASM'
                                <1>
                                <1>       ; enable codec, unmute stuff, set output rate
                                <1> ;     ; entry: [audio_freq] = desired sample rate
                                <1>
                                <1> ;     mov    dx, [NAMBAR]
                                <1> ;     add    dx, CODEC_EXT_AUDIO_CTRL_REG ; 2Ah
                                <1> ;     in     ax, dx
                                <1> ;     or     ax, 1
                                <1> ;     out    dx, ax                   ; Enable variable rate audio
                                <1>
                                <1> ;     ;call  delay1_4ms
                                <1> ;     ;call  delay1_4ms
                                <1> ;     ;call  delay1_4ms
                                <1> ;     ;call  delay1_4ms
                                <1>
                                <1> ;     mov    ax, [audio_freq]         ; sample rate
                                <1>
                                <1> ;     mov    dx, [NAMBAR]
                                <1> ;     add    dx, CODEC_PCM_FRONT_DACRATE_REG  ; 2Ch
                                <1> ;     out    dx, ax                   ; out sample rate
                                <1>
                                <1> ;     ;call  delay1_4ms
                                <1> ;     ;call  delay1_4ms
                                <1> ;     ;call  delay1_4ms
                                <1> ;     ;call  delay1_4ms
                                <1>
                                <1>       ;mov   dx, [NAMBAR]              ; mixer base address
                                <1>       ;add   dx, CODEC_RESET_REG       ; reset register
                                <1>       ;mov   ax, 42
                                <1>       ;out   dx, ax                   ; reset
                                <1>
                                <1>       ;mov    dx, [NABMBAR]            ; bus master base address
                                <1>       ;add   dx, GLOB_STS_REG
                                <1>       ;mov   ax, 2
                                <1>       ;out   dx, ax
                                <1>
00014D07 E847F9FFFF             <1>       call   delay_100ms ; 29/05/2017
                                <1>
                                <1> init_ac97_codec:
                                <1>       ; 10/06/2017
                                <1>       ; 29/05/2017
                                <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
                                <1>       ;
```

```
5500 00014D0C 66BA2C00            <1>       mov   dx, GLOB_CNT_REG ; 2Ch
5501 00014D10 660315[12900100]    <1>       add   dx, [NABMBAR]
5502 00014D17 ED                  <1>       in    eax, dx
5503                              <1>       ; ?
5504 00014D18 66BA3000            <1>       mov   dx, GLOB_STS_REG ; 30h
5505 00014D1C 660315[12900100]    <1>       add   dx, [NABMBAR]
5506 00014D23 ED                  <1>       in    eax, dx
5507                              <1>
5508 00014D24 83F8FF              <1>       cmp   eax, 0FFFFFFFFh ; -1
5509 00014D27 744B                <1>       je    short init_ac97_codec_err1
5510                              <1>
5511 00014D29 A900030010          <1>       test  eax, CTRL_ST_CREADY
5512 00014D2E 7507                <1>       jnz   short _ac97_codec_ready
5513                              <1>
5514 00014D30 E8EF020000          <1>       call  reset_ac97_codec
5515 00014D35 723E                <1>       jc    short init_ac97_codec_err2
5516                              <1>
5517                              <1> _ac97_codec_ready:
5518 00014D37 668B15[10900100]    <1>       mov   dx, [NAMBAR]
5519                              <1>       ;add  dx, 0 ; ac_reg_0 ; reset register
5520 00014D3E 66EF                <1>       out   dx, ax
5521                              <1>
5522 00014D40 31C0                <1>       xor   eax, eax ; 0
5523 00014D42 668B15[10900100]    <1>       mov   dx, [NAMBAR]
5524 00014D49 6683C226            <1>       add   dx, CODEC_REG_POWERDOWN
5525 00014D4D 66EF                <1>       out   dx, ax
5526                              <1>
5527                              <1>       ; 10/06/2017
5528                              <1>       ; 29/05/2017
5529                              <1>       ; wait for 1 second
5530 00014D4F B9E8030000          <1>       mov   ecx, 1000 ; 1000*0.25ms = 1s
5531                              <1> _ac97_codec_rloop:
5532 00014D54 E807F9FFFF          <1>       call  delay1_4ms
5533 00014D59 E802F9FFFF          <1>       call  delay1_4ms
5534 00014D5E E8FDF8FFFF          <1>       call  delay1_4ms
5535 00014D63 E8F8F8FFFF          <1>       call  delay1_4ms
5536                              <1>       ;mov  dx, [NAMBAR]
5537                              <1>       ;add  dx, CODEC_REG_POWERDOWN
5538 00014D68 66ED                <1>       in    ax, dx
5539 00014D6A 6683E00F            <1>       and   ax, 0Fh
5540 00014D6E 3C0F                <1>       cmp   al, 0Fh
5541 00014D70 7404                <1>       je    short _ac97_codec_init_ok
5542 00014D72 E2E0                <1>       loop  _ac97_codec_rloop
5543                              <1>
5544                              <1> init_ac97_codec_err1:
5545 00014D74 F9                  <1>       stc
5546                              <1> init_ac97_codec_err2:
5547 00014D75 C3                  <1>       retn
5548                              <1>
5549                              <1> _ac97_codec_init_ok:
5550 00014D76 B002                <1>         mov    al, 2 ; force set 16-bit 2-channel PCM
5551 00014D78 66BA2C00            <1>       mov   dx, GLOB_CNT_REG ; 2Ch
5552 00014D7C 660315[12900100]    <1>       add   dx, [NABMBAR]
5553 00014D83 EF                  <1>       out   dx, eax
5554                              <1>
5555                              <1>       ;call  delay1_4ms
5556                              <1>
5557                              <1>       ; 10/06/2017
5558 00014D84 E849020000          <1>       call  reset_ac97_controller
5559                              <1>
5560                              <1> ;     call  setup_ac97_codec
5561                              <1> ;
5562                              <1> ;detect_ac97_codec:
5563                              <1> ;     retn
5564                              <1>
5565                              <1> setup_ac97_codec:
5566                              <1>       ; 22/07/2020
5567                              <1>       ; 10/06/2017
5568                              <1>       ; 29/05/2017
5569 00014D89 B802020000          <1>       mov   eax, 0202h
5570 00014D8E 66A3[0E900100]      <1>       mov   [audio_master_volume], ax
5571 00014D94 66B81F1F            <1>       mov   ax, 1F1Fh ; 31, 31
5572                              <1>
5573 00014D98 668B15[10900100]    <1>       mov   dx, [NAMBAR]
5574 00014D9F 6683C202            <1>       add   dx, CODEC_MASTER_VOL_REG   ;02h
5575 00014DA3 6631C0              <1>       xor   ax, ax       ; volume attenuation = 0 (max. volume)
5576 00014DA6 66EF                <1>       out   dx, ax
5577                              <1>
5578 00014DA8 668B15[10900100]    <1>       mov   dx, [NAMBAR]
5579 00014DAF 6683C206            <1>       add   dx, CODEC_MASTER_MONO_VOL_REG    ;06h
5580                              <1>       ;xor  ax, ax
5581 00014DB3 66EF                <1>       out   dx, ax
5582                              <1>
5583 00014DB5 668B15[10900100]    <1>       mov   dx, [NAMBAR]
5584 00014DBC 6683C20A            <1>       add   dx, CODEC_PCBEEP_VOL_REG   ;0Ah
5585                              <1>       ;xor  ax, ax
5586 00014DC0 66EF                <1>       out   dx, ax
5587                              <1>
5588 00014DC2 668B15[10900100]    <1>       mov   dx, [NAMBAR]
5589 00014DC9 6683C218            <1>       add   dx, CODEC_PCM_OUT_REG            ;18h
5590                              <1>       ;xor  ax, ax
5591 00014DCD 66EF                <1>       out   dx, ax
5592                              <1>
5593 00014DCF 66B80880            <1>       mov   ax, 8008h ; Mute
5594 00014DD3 668B15[10900100]    <1>       mov   dx, [NAMBAR]
5595                              <1>       ; 22/07/2020
5596 00014DDA 6683C20C            <1>       add    dx, CODEC_PHONE_VOL_REG          ;0Ch
5597                              <1>                           ; AC97_PHONE_VOL ; TAD Input (Mono)
5598 00014DDE 66EF                <1>       out    dx, ax
5599                              <1>
5600 00014DE0 66B80808            <1>       mov    ax, 0808h
5601 00014DE4 668B15[10900100]    <1>       mov   dx, [NAMBAR]
5602 00014DEB 6683C210            <1>       add    dx, CODEC_LINE_IN_VOL_REG ;10h ; Line Input (Stereo)
5603 00014DEF 66EF                <1>       out   dx, ax
5604                              <1>
```

```
                                <1>       ;mov   ax, 0808h
00014DF1 668B15[10900100]       <1>       mov   dx, [NAMBAR]
00014DF8 6683C212               <1>       add    dx, CODEC_CD_VOL_REG ;12h ; CR Input (Stereo)
00014DFC 66EF                   <1>       out   dx, ax
                                <1>
                                <1>       ;mov   ax, 0808h
00014DFE 668B15[10900100]       <1>       mov   dx, [NAMBAR]
00014E05 6683C216               <1>       add    dx, CODEC_AUX_VOL_REG ;16h ; Aux Input (Stereo)
00014E09 66EF                   <1>       out   dx, ax
                                <1>
                                <1>      ;call  delay1_4ms
                                <1>      ;call  delay1_4ms
                                <1>      ;call  delay1_4ms
                                <1>      ;call  delay1_4ms
                                <1>
                                <1> detect_ac97_codec:
00014E0B C3                     <1>       retn
                                <1>
                                <1> set_ac97_bdl: ; Set AC97 (ICH) Buffer Descriptor List
                                <1>       ; 17/06/2017
                                <1>       ; 11/06/2017
                                <1>       ; 28/05/2017
                                <1>       ; eax = dma buffer address = [audio_DMA_buff]
                                <1>       ; ecx = dma buffer buffer size = [audio_dmabuff_size]
                                <1>
00014E0C D1E9                   <1>       shr    ecx, 1 ; dma half buffer size
00014E0E 89CE                   <1>       mov    esi, ecx
                                <1>
00014E10 BF[14900100]           <1>       mov    edi, audio_bdl_buff ; get BDL address
00014E15 B910000000             <1>       mov    ecx, 32 / 2         ; make 32 entries in BDL
                                <1>
00014E1A EB05                   <1>       jmp    short s_ac97_bdl1
                                <1>
                                <1> s_ac97_bdl0:
                                <1>       ; set buffer descriptor 0 to start of data file in memory
                                <1>
00014E1C A1[F88F0100]           <1>       mov    eax, [audio_dma_buff]     ; Physical address of DMA buffer
                                <1>
                                <1> s_ac97_bdl1:
00014E21 AB                     <1>       stosd                    ; store dmabuffer1 address
                                <1>
00014E22 89C2                   <1>       mov    edx, eax
                                <1>
                                <1> ;
                                <1> ; Buffer Descriptors List
                                <1> ; As stated earlier, each buffer descriptor list is a set of (up to) 32
                                <1> ; descriptors, each 8 bytes in length. Bytes 0-3 of a descriptor entry point
                                <1> ; to a chunk of memory to either play from or record to. Bytes 4-7 of an
                                <1> ; entry describe various control things detailed below.
                                <1> ;
                                <1> ; Buffer pointers must always be aligned on a Dword boundry.
                                <1> ;
                                <1> ;
                                <1>
                                <1> ;IOC        equ     BIT31       ; Fire an interrupt whenever this
                                <1>                                 ; buffer is complete.
                                <1>
                                <1> ;BUP        equ     BIT30       ; Buffer Underrun Policy.
                                <1>                                 ; if this buffer is the last buffer
                                <1>                                 ; in a playback, fill the remaining
                                <1>                                 ; samples with 0 (silence) or not.
                                <1>                                 ; It's a good idea to set this to 1
                                <1>                                 ; for the last buffer in playback,
                                <1>                                 ; otherwise you're likely to get a lot
                                <1>                                 ; of noise at the end of the sound.
                                <1>
                                <1> ;
                                <1> ; Bits 15:0 contain the length of the buffer, in number of samples, which
                                <1> ; are 16 bits each, coupled in left and right pairs, or 32bits each.
                                <1> ; Luckily for us, that's the same format as .wav files.
                                <1> ;
                                <1> ; A value of FFFF is 65536 samples. Running at 44.1Khz, that's just about
                                <1> ; 1.5 seconds of sample time. FFFF * 32bits is 1FFFFh bytes or 128k of data.
                                <1> ;
                                <1> ; A value of 0 in these bits means play no samples.
                                <1> ;
                                <1>
00014E24 89F0                   <1>       mov    eax, esi ; DMA half buffer size
00014E26 01C2                   <1>       add    edx, eax
00014E28 D1E8                   <1>       shr    eax, 1 ; count of 16 bit samples
                                <1>       ;or    eax, IOC+BUP
00014E2A 0D00000080             <1>       or     eax, IOC ; 11/06/2017
00014E2F AB                     <1>       stosd
                                <1>
                                <1> ; 2nd buffer:
                                <1>
00014E30 89D0                   <1>       mov    eax, edx ; Physical address of the 2nd half of DMA buffer
00014E32 AB                     <1>       stosd          ; store dmabuffer2 address
                                <1>
                                <1> ; set length to [audio_dmabuff_size]/2
                                <1> ; Set control (bits 31:16) to BUP, bits 15:0=number of samples
                                <1> ;
00014E33 89F0                   <1>       mov    eax, esi ; DMA half buffer size
00014E35 D1E8                   <1>       shr    eax, 1 ; count of 16 bit samples
                                <1>       ;or    eax, IOC+BUP
00014E37 0D00000080             <1>       or     eax, IOC ; 11/06/2017
00014E3C AB                     <1>       stosd
                                <1>
00014E3D E2DD                   <1>       loop    s_ac97_bdl0
                                <1>
00014E3F C3                     <1>       retn
                                <1>
                                <1> ac97_start_play:
                                <1>       ; 28/05/2017
                                <1>       ; Derived from 'playWav' procedure in 'ICHWAV.ASM'
```

```
5710                                    <1>       ; .wav player for DOS by Jeff Leyda (02/09/2002)
5711                                    <1>
5712                                    <1>       ; set output rate
5713                                    <1>       ; entry: [audio_freq] = desired sample rate
5714                                    <1>
5715 00014E40 668B15[10900100]         <1>       mov    dx, [NAMBAR]
5716 00014E47 6683C22A                 <1>       add    dx, CODEC_EXT_AUDIO_CTRL_REG     ; 2Ah
5717 00014E4B 66ED                     <1>       in     ax, dx
5718 00014E4D 6683C801                 <1>       or     ax, 1
5719 00014E51 66EF                     <1>       out    dx, ax                           ; Enable variable rate audio
5720                                    <1>
5721                                    <1>      ;call   delay1_4ms
5722                                    <1>      ;call   delay1_4ms
5723                                    <1>      ;call   delay1_4ms
5724                                    <1>      ;call   delay1_4ms
5725                                    <1>
5726 00014E53 66A1[0A900100]           <1>       mov    ax, [audio_freq]                 ; sample rate
5727                                    <1>
5728 00014E59 668B15[10900100]         <1>       mov    dx, [NAMBAR]
5729 00014E60 6683C22C                 <1>       add    dx, CODEC_PCM_FRONT_DACRATE_REG  ; 2Ch
5730 00014E64 66EF                     <1>       out    dx, ax                           ; out sample rate
5731                                    <1>
5732                                    <1>       ;call  delay1_4ms
5733                                    <1>       ;call  delay1_4ms
5734                                    <1>       ;call  delay1_4ms
5735                                    <1>       ;call  delay1_4ms
5736                                    <1>
5737                                    <1> ;
5738                                    <1> ; register reset the DMA engine. This may cause a pop noise on the output
5739                                    <1> ; lines when the device is reset. Prolly a better idea to mute output, then
5740                                    <1> ; reset.
5741                                    <1> ;
5742 00014E66 668B15[12900100]         <1>       mov    dx, [NABMBAR]
5743 00014E6D 6683C21B                 <1>       add    dx, PO_CR_REG                    ; set pointer to Cntl reg
5744 00014E71 B002                     <1>       mov    al, RR                           ; set reset
5745 00014E73 EE                       <1>       out    dx, al                           ; self clearing bit
5746                                    <1> ;
5747                                    <1> ;     mov    edi, audio_bdl_buff
5748                                    <1> ;     mov    edx, [audio_dmabuff_size]
5749                                    <1> ;     shr    edx, 1
5750                                    <1> ;     mov    ecx, 32/2
5751                                    <1> ;ac97_set_bdl_buffer:
5752                                    <1> ;     ; 1st half of DMA buffer
5753                                    <1> ;     mov    eax, [audio_dma_buff]
5754                                    <1> ;     push   eax
5755                                    <1> ;     stosd
5756                                    <1> ;     mov    eax, edx ; dma buffer size / 2
5757                                    <1> ;     or     eax, IOC+BUP
5758                                    <1> ;     stosd
5759                                    <1> ;     pop    eax
5760                                    <1> ;     ; 2nd half of DMA buffer
5761                                    <1> ;     add    eax, edx
5762                                    <1> ;     stosd
5763                                    <1> ;     mov    eax, edx ; dma buffer size / 2
5764                                    <1> ;     or     eax, IOC+BUP
5765                                    <1> ;     stosd
5766                                    <1> ;     loop   ac97_set_bdl_buffer
5767                                    <1>
5768                                    <1> ; tell the DMA engine where to find our list of Buffer Descriptors.
5769                                    <1> ; this 32bit value is a flat mode memory offset (ie no segment:offset)
5770                                    <1> ;
5771                                    <1> ; write NABMBAR+10h with offset of buffer descriptor list
5772                                    <1> ;
5773 00014E74 B8[14900100]              <1>       mov    eax, audio_bdl_buff
5774 00014E79 668B15[12900100]         <1>       mov    dx, [NABMBAR]
5775 00014E80 6683C210                 <1>       add    dx, PO_BDBAR_REG
5776 00014E84 EF                       <1>       out    dx, eax
5777                                    <1> ;
5778                                    <1> ; All set. Let's play some music.
5779                                    <1> ;
5780                                    <1> ;
5781 00014E85 B81F000000               <1>       mov    eax, 31
5782 00014E8A E816000000               <1>       call   set_ac97_LastValidIndex
5783                                    <1>
5784 00014E8F C605[0C900100]01         <1>       mov    byte [audio_play_cmd], 1 ; play command (do not stop) !
5785                                    <1>
5786                                    <1> ac97_play: ; continue to play (after pause)
5787                                    <1>       ; 11/06/2017
5788                                    <1>       ; 29/05/2017
5789                                    <1>       ; 28/05/2017
5790 00014E96 668B15[12900100]         <1>       mov    dx, [NABMBAR]
5791 00014E9D 6683C21B                 <1>       add    dx, PO_CR_REG       ; PCM out control register
5792 00014EA1 B011                     <1>       mov    al, IOCE+RPBM ; 29/05/2017
5793                                    <1>       ;mov   al, 1Dh      ; (Ref: KolibriOS, intelac97.asm, 'play:')
5794 00014EA3 EE                       <1>       out    dx, al              ; set start!
5795                                    <1>
5796                                    <1>       ;mov   byte [audio_play_cmd], 1 ; play command (do not stop) !
5797                                    <1>
5798 00014EA4 C3                       <1>       retn
5799                                    <1>
5800                                    <1> ;input AL = index # to stop on
5801                                    <1> set_ac97_LastValidIndex:
5802                                    <1>       ; 28/05/2017
5803                                    <1>       ; Derived from 'setLastValidIndex' procedure in 'ICHWAV.ASM'
5804                                    <1>       ; .wav player for DOS by Jeff Leyda (02/09/2002)
5805 00014EA5 668B15[12900100]         <1>       mov    dx, [NABMBAR]
5806 00014EAC 6683C215                 <1>       add    dx, PO_LVI_REG
5807 00014EB0 EE                       <1>       out    dx, al
5808                                    <1>       ;mov   [audio_lvi], al ; for ac97_int_handler
5809 00014EB1 C3                       <1>       retn
5810                                    <1>
5811                                    <1> ac97_volume:
5812                                    <1>       ; 28/05/2017
5813                                    <1>       ; bl = component (0 = master/playback/lineout volume)
5814                                    <1>       ; cl = left channel volume level (0 to 31)
```

```
                                <1>       ; ch = right channel volume level (0 to 31)
                                <1>
00014EB2 08DB                   <1>       or     bl, bl
00014EB4 7523                   <1>       jnz    short ac97_vol_1 ; temporary !
00014EB6 66B81F1F               <1>       mov    ax, 1F1Fh ; 31,31
00014EBA 38C1                   <1>       cmp    cl, al
00014EBC 771B                   <1>       ja     short ac97_vol_1 ; temporary !
00014EBE 38E5                   <1>       cmp    ch, ah
00014EC0 7717                   <1>       ja     short ac97_vol_1 ; temporary !
00014EC2 66890D[0E900100]       <1>       mov    [audio_master_volume], cx
00014EC9 6629C8                 <1>       sub    ax, cx
00014ECC 668B15[10900100]       <1>       mov    dx, [NAMBAR]
00014ED3 6683C202               <1>       add    dx, CODEC_MASTER_VOL_REG  ; 02h ; Line Out
00014ED7 66EF                   <1>       out    dx, ax
                                <1> ac97_vol_1:
00014ED9 C3                     <1>       retn
                                <1>
                                <1> ac97_int_handler:
                                <1>       ; 12/10/2017
                                <1>       ; 10/10/2017
                                <1>       ; 09/10/2017
                                <1>       ; 13/06/2017, 13/06/2017
                                <1>       ; 10/06/2017, 11/06/2017
                                <1>       ; Interrupt Handler for AC97 (ICH) Audio Controller
                                <1>       ; Note: called by 'dev_IRQ_service'
                                <1>       ; 28/05/2017
                                <1>
                                <1>       ;push  eax ; * must be saved !
                                <1>       ;push  edx
                                <1>       ;push  ecx
                                <1>       ;push  ebx ; * must be saved !
                                <1>       ;push  esi
                                <1>       ;push  edi
                                <1>
                                <1>       ;cmp   byte [audio_busy], 1
                                <1>       ;jnb   _ac97_ih2    ; busy !
                                <1>
00014EDA 66BA3000               <1>       mov    dx, GLOB_STS_REG
00014EDE 660315[12900100]       <1>       add    dx, [NABMBAR]
00014EE5 ED                     <1>       in     eax, dx
                                <1>
00014EE6 83F8FF                 <1>       cmp    eax, 0FFFFFFFFh ; -1
00014EE9 0F849A000000           <1>       je     _ac97_ih3 ; exit
                                <1>
00014EEF A940000000             <1>       test   eax, 40h ; PCM Out Interrupt
00014EF4 750E                   <1>       jnz    short _ac97_ih0
                                <1>
00014EF6 85C0                   <1>       test   eax, eax
00014EF8 0F848B000000           <1>       jz     _ac97_ih3 ; exit
                                <1>
                                <1>       ;mov   dx, GLOB_STS_REG
                                <1>       ;add   dx, [NABMBAR]
00014EFE EF                     <1>       out    dx, eax
                                <1>
00014EFF E985000000             <1>       jmp    _ac97_ih3 ; exit
                                <1>
                                <1> _ac97_ih0:
00014F04 50                     <1>       push   eax
                                <1>       ; 09/10/2017
00014F05 803D[0C900100]01       <1>       cmp    byte [audio_play_cmd], 1
00014F0C 727C                   <1>       jb     short _ac97_ih4 ; stop command !
                                <1>
                                <1>       ;mov   byte [audio_busy], 1
                                <1>
                                <1>       ;mov   al, 10h
                                <1>       ;mov   dx, PO_CR_REG
                                <1>       ;add   dx, [NABMBAR]
                                <1>       ;out   dx, al
                                <1>
00014F0E 66B81C00               <1>       mov    ax, 1Ch ; FIFOE(=16)+BCIS(=8)+LVBCI(=4)
00014F12 66BA1600               <1>       mov    dx, PO_SR_REG
00014F16 660315[12900100]       <1>       add    dx, [NABMBAR]
00014F1D 66EF                   <1>       out    dx, ax
                                <1>
00014F1F 66BA1400               <1>       mov    dx, PO_CIV_REG
00014F23 660315[12900100]       <1>       add    dx, [NABMBAR]
00014F2A EC                     <1>       in     al, dx
                                <1>
                                <1>       ;cmp   al, [audio_civ] ; [audio_flag]
                                <1>       ;je    short _ac97_ih2
                                <1>
00014F2B A2[0D900100]           <1>       mov    [audio_civ], al
00014F30 FEC8                   <1>       dec    al
                                <1>       ;inc   al ; 11/06/2017
00014F32 241F                   <1>       and    al, 1Fh
                                <1>
00014F34 66BA1500               <1>       mov     dx, PO_LVI_REG
00014F38 660315[12900100]       <1>       add    dx, [NABMBAR]
00014F3F EE                     <1>       out    dx, al
                                <1>
                                <1>       ; 12/10/2017
00014F40 A0[0D900100]           <1>       mov    al, [audio_civ]
00014F45 FEC0                   <1>       inc    al
00014F47 2401                   <1>       and    al, 1
00014F49 A2[00900100]           <1>       mov    [audio_flag], al
                                <1>       ;; [audio_flag] : 0 = Buffer 1, 1 = Buffer 2
                                <1>       ;
00014F4E 58                     <1>       pop    eax
                                <1>       ;
00014F4F 83E040                 <1>       and    eax, 40h
00014F52 668B15[12900100]       <1>       mov    dx, [NABMBAR]
00014F59 6683C230               <1>       add    dx, GLOB_STS_REG
00014F5D EF                     <1>       out    dx, eax
                                <1>
                                <1>       ;; 13/06/2017
```

```
                            <1>       ;mov   al, 11h ; IOCE + RPBM
                            <1>       ;mov   dx, PO_CR_REG
                            <1>       ;add   dx, [NABMBAR]
                            <1>       ;out   dx, al
                            <1>
                            <1> ac97_tuneloop:
                            <1>       ; 09/10/2017
00014F5E 8B3D[F88F0100]     <1>       mov    edi, [audio_dma_buff]
00014F64 8B0D[FC8F0100]     <1>       mov    ecx, [audio_dmabuff_size]
00014F6A D1E9               <1>       shr    ecx, 1 ; dma buff size / 2 = half buffer size
                            <1>
                            <1>       ; 12/10/2017
00014F6C 803D[00900100]00   <1>       cmp    byte [audio_flag], 0
00014F73 7702               <1>       ja     short _ac97_ih1  ; Playing Half Buffer 2 (Current: FLAG)
                            <1>       ; Playing Half Buffer 1 (Current: EOL)
00014F75 01CF               <1>       add    edi, ecx
                            <1> _ac97_ih1:
                            <1>       ; Update half buffer 2 while playing half buffer 1 (next: FLAG)
                            <1>       ; Update half buffer 1 while playing half buffer 2 (next: EOL)
                            <1>
00014F77 8B35[F08F0100]     <1>       mov    esi, [audio_p_buffer] ; phy addr of audio buff
00014F7D C1E902             <1>       shr    ecx, 2 ; half buff size / 4
00014F80 F3A5               <1>       rep    movsd
                            <1>
                            <1>       ; 10/10/2017
                            <1>       ; switch flag value
00014F82 8035[00900100]01   <1>       xor    byte [audio_flag], 1
                            <1>       ; 12/10/2017
                            <1>       ; [audio_flag] = 0 : Playing dma half buffer 2 (even index value)
                            <1>                          ; Next buffer (to update) is dma half buff 1
                            <1>       ;            = 1 : Playing dma half buffer 1 (odd index value)
                            <1>                          ; Next buffer (to update) is dma half buff 2
                            <1>
                            <1> _ac97_ih2:
                            <1>       ;mov   byte [audio_busy], 0
                            <1> _ac97_ih3:
                            <1>       ;pop   edi
                            <1>       ;pop   esi
                            <1>       ;pop   ebx ; * must be restored !
                            <1>       ;pop   ecx
                            <1>       ;pop   edx
                            <1>       ;pop   eax ; * must be restored !
                            <1>
00014F89 C3                 <1>       retn
                            <1>
                            <1> _ac97_ih4:
                            <1>       ; 09/10/2017
00014F8A E818000000         <1>       call   _ac97_stop
                            <1>       ;
00014F8F 58                 <1>       pop    eax
                            <1>       ;
00014F90 83E040             <1>       and    eax, 40h
00014F93 668B15[12900100]   <1>       mov    dx, [NABMBAR]
00014F9A 6683C230           <1>       add    dx, GLOB_STS_REG
00014F9E EF                 <1>       out    dx, eax
                            <1>
                            <1>       ;; 13/06/2017
                            <1>       ;mov   al, 11h ; IOCE + RPBM
                            <1>       ;add dx, PO_CR_REG
                            <1>       ;add   dx, [NABMBAR]
                            <1>       ;out   dx, al
                            <1>
                            <1>       ; 10/10/2017
                            <1>       ;jmp   short _ac97_ih3  ; exit
00014F9F C3                 <1>       retn
                            <1>
                            <1> ac97_stop:
                            <1>       ; 28/05/2017
00014FA0 C605[0C900100]00   <1>       mov    byte [audio_play_cmd], 0 ; stop !
                            <1> _ac97_stop: ; 09/10/2017
                            <1>       ; 29/05/2017
                            <1>       ;mov   dx, [NABMBAR]
                            <1>       ;add   dx, PO_CR_REG
                            <1>       ;mov   al, 0
                            <1>       ;out   dx, al
                            <1>
                            <1>       ; 11/06/2017
00014FA7 30C0               <1>       xor    al, al ; 0
00014FA9 E813000000         <1>       call   ac97_po_cmd
                            <1>
                            <1>       ; (Ref: KolibriOS, intelac97.asm, 'stop:')
                            <1>       ; Clear FIFOE, BCIS, LVBCI (Ref: Intel ICH hub manual)
00014FAE 66B81C00           <1>       mov    ax, 1Ch
00014FB2 668B15[12900100]   <1>       mov    dx, [NABMBAR]
00014FB9 6683C216           <1>       add    dx, PO_SR_REG
00014FBD 66EF               <1>       out    dx, ax
                            <1>
                            <1>       ;retn
                            <1>
                            <1>       ; 11/06/2017
00014FBF B002               <1>       mov    al, RR
                            <1> ac97_po_cmd:
                            <1>        ;11/06/2017
                            <1>       ; 29/05/2017
00014FC1 668B15[12900100]   <1>       mov    dx, [NABMBAR]
00014FC8 6683C21B           <1>       add    dx, PO_CR_REG       ; PCM out control register
00014FCC EE                 <1>       out    dx, al
00014FCD C3                 <1>       retn
                            <1>
                            <1> ac97_pause:
                            <1>       ; 11/06/2017
                            <1>       ; 29/05/2017
00014FCE B010               <1>       mov    al, IOCE
00014FD0 EBEF               <1>       jmp    short ac97_po_cmd
                            <1>
```

```
                            <1> reset_ac97_controller:
                            <1>       ; 10/06/2017
                            <1>       ; 29/05/2017
                            <1>       ; 28/05/2017
                            <1>       ; reset AC97 audio controller registers
00014FD2 31C0               <1>       xor    eax, eax
00014FD4 66BA0B00           <1>       mov    dx, PI_CR_REG
00014FD8 660315[12900100]   <1>       add    dx, [NABMBAR]
00014FDF EE                 <1>       out    dx, al
                            <1>
00014FE0 66BA1B00           <1>       mov    dx, PO_CR_REG
00014FE4 660315[12900100]   <1>       add    dx, [NABMBAR]
00014FEB EE                 <1>       out    dx, al
                            <1>
00014FEC 66BA2B00           <1>       mov    dx, MC_CR_REG
00014FF0 660315[12900100]   <1>       add    dx, [NABMBAR]
00014FF7 EE                 <1>       out    dx, al
                            <1>
00014FF8 B002               <1>       mov    al, RR
00014FFA 66BA0B00           <1>       mov    dx, PI_CR_REG
00014FFE 660315[12900100]   <1>       add    dx, [NABMBAR]
00015005 EE                 <1>       out    dx, al
                            <1>
00015006 66BA1B00           <1>       mov    dx, PO_CR_REG
0001500A 660315[12900100]   <1>       add    dx, [NABMBAR]
00015011 EE                 <1>       out    dx, al
                            <1>
00015012 66BA2B00           <1>       mov    dx, MC_CR_REG
00015016 660315[12900100]   <1>       add    dx, [NABMBAR]
0001501D EE                 <1>       out    dx, al
                            <1>
0001501E C3                 <1>       retn
                            <1>
                            <1> ac97_reset:
                            <1>       ; 10/06/2017
                            <1>       ; 29/05/2017
                            <1>       ; 28/05/2017
0001501F E8AEFFFFFF         <1>       call   reset_ac97_controller
                            <1>       ; 29/05/2017
                            <1>       ;jmp   reset_ac97_codec
                            <1> reset_ac97_codec:
                            <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
00015024 66BA2C00           <1>       mov    dx, GLOB_CNT_REG ; 2Ch
00015028 660315[12900100]   <1>       add    dx, [NABMBAR]
0001502F ED                 <1>       in     eax, dx
                            <1>
00015030 A902000000         <1>       test   eax, 2
00015035 7407               <1>       jz     short _r_ac97codec_cold
                            <1>
00015037 E80F000000         <1>       call   warm_ac97codec_reset
0001503C 7308               <1>       jnc    short _r_ac97codec_ok
                            <1> _r_ac97codec_cold:
0001503E E83D000000         <1>       call   cold_ac97codec_reset
00015043 7301               <1>       jnc    short _r_ac97codec_ok
                            <1>
                            <1>       ; 16/04/2017
                            <1>       ;xor  eax, eax         ; timeout error
                            <1>       ;stc
00015045 C3                 <1>       retn
                            <1>
                            <1> _r_ac97codec_ok:
00015046 31C0               <1>       xor    eax, eax
                            <1>       ;mov   al, VIA_ACLINK_C00_READY ; 1
00015048 FEC0               <1>       inc    al
0001504A C3                 <1>       retn
                            <1>
                            <1> warm_ac97codec_reset:
                            <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
0001504B B806000000         <1>       mov    eax, 6
00015050 66BA2C00           <1>       mov    dx, GLOB_CNT_REG ; 2Ch
00015054 660315[12900100]   <1>       add    dx, [NABMBAR]
0001505B EF                 <1>       out    dx, eax
                            <1>
0001505C B90A000000         <1>       mov    ecx, 10      ; total 1s
                            <1> _warm_ac97c_rst_wait:
00015061 51                 <1>       push   ecx
00015062 E8ECF5FFFF         <1>       call   delay_100ms
00015067 59                 <1>       pop    ecx
                            <1>
00015068 66BA3000           <1>       mov    dx, GLOB_STS_REG ; 30h
0001506C 660315[12900100]   <1>       add    dx, [NABMBAR]
00015073 ED                 <1>       in     eax, dx
                            <1>
00015074 A900030010         <1>       test   eax, CTRL_ST_CREADY
00015079 7504               <1>       jnz    short _warm_ac97c_rst_ok
                            <1>
0001507B 49                 <1>       dec    ecx
0001507C 75E3               <1>       jnz    short _warm_ac97c_rst_wait
                            <1>
                            <1> _warm_ac97c_rst_fail:
0001507E F9                 <1>       stc
                            <1> _warm_ac97c_rst_ok:
0001507F C3                 <1>       retn
                            <1>
                            <1> cold_ac97codec_reset:
                            <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
00015080 B802000000         <1>       mov    eax, 2
00015085 66BA2C00           <1>       mov    dx, GLOB_CNT_REG ; 2Ch
00015089 660315[12900100]   <1>       add    dx, [NABMBAR]
00015090 EF                 <1>       out    dx, eax
                            <1>
00015091 E8BDF5FFFF         <1>       call   delay_100ms  ; wait 100 ms
00015096 E8B8F5FFFF         <1>       call   delay_100ms  ; wait 100 ms
0001509B E8B3F5FFFF         <1>       call   delay_100ms  ; wait 100 ms
000150A0 E8AEF5FFFF         <1>       call   delay_100ms  ; wait 100 ms
```

```
6130                                    <1>
6131 000150A5 B910000000                <1>     mov     ecx, 16         ; total 20*100 ms = 2s
6132                                    <1> _cold_ac97c_rst_wait:
6133 000150AA 66BA3000                  <1>     mov     dx, GLOB_STS_REG ; 30h
6134 000150AE 660315[12900100]          <1>     add     dx, [NABMBAR]
6135 000150B5 ED                        <1>     in      eax, dx
6136                                    <1>
6137 000150B6 A900030010                <1>     test    eax, CTRL_ST_CREADY
6138 000150BB 750B                      <1>     jnz     short _cold_ac97c_rst_ok
6139                                    <1>
6140 000150BD 51                        <1>     push    ecx
6141 000150BE E890F5FFFF                <1>     call    delay_100ms
6142 000150C3 59                        <1>     pop     ecx
6143                                    <1>
6144 000150C4 49                        <1>     dec     ecx
6145 000150C5 75E3                      <1>     jnz     short _cold_ac97c_rst_wait
6146                                    <1>
6147                                    <1> _cold_ac97c_rst_fail:
6148 000150C7 F9                        <1>       stc
6149                                    <1> _cold_ac97c_rst_ok:
6150 000150C8 C3                        <1>     retn
6151                                    <1>
6152                                    <1> sb16_current_sound_data:
6153                                    <1>     ; 20/08/2017
6154                                    <1>     ; 24/06/2017
6155                                    <1>     ; 22/06/2017
6156                                    <1>     ; get current sound (PCM out) data for graphics
6157                                    <1>     ; (for Sound Blaster 16)
6158                                    <1>     ; ebx = Physical address (on page boundary)
6159                                    <1>     ; ecx = Byte count
6160                                    <1>     ; [audio_buff_size]
6161                                    <1>
6162                                    <1>     ;;mov  edi, [audio_buff_size]
6163                                    <1>     ;mov   edi, [audio_dmabuff_size]
6164                                    <1>     ;mov   esi, [audio_dma_buff]
6165 000150C9 39CF                      <1>     cmp    edi, ecx
6166 000150CB 7302                      <1>     jnb    short sb16_gcd_0
6167 000150CD 89F9                      <1>     mov    ecx, edi
6168                                    <1> sb16_gcd_0:
6169                                    <1>     ; 20/08/2017
6170 000150CF 803D[08900100]10          <1>     cmp    byte [audio_bps], 16
6171 000150D6 750F                      <1>     jne    short sb16_gcd_1 ; 8 bit DMA channel
6172 000150D8 E4C6                      <1>     in     al, 0C6h ; DMA channel 5 count register
6173 000150DA 88C2                      <1>     mov    dl, al
6174 000150DC E4C6                      <1>     in     al, 0C6h
6175 000150DE 88C6                      <1>     mov    dh, al
6176 000150E0 0FB7C2                    <1>     movzx  eax, dx
6177 000150E3 D1E0                      <1>     shl    eax, 1 ; word count -> byte count
6178 000150E5 EB4E                      <1>     jmp    short sb16_gcd_2
6179                                    <1> sb16_gcd_1:
6180 000150E7 E403                      <1>     in     al, 03h ; DMA channel 1 count register
6181 000150E9 88C2                      <1>     mov    dl, al
6182 000150EB E403                      <1>     in     al, 03h
6183 000150ED 88C6                      <1>     mov    dh, al
6184 000150EF 0FB7C2                    <1>     movzx  eax, dx
6185 000150F2 EB41                      <1>     jmp    short sb16_gcd_2
6186                                    <1> ;sb16_gcd_2:
6187                                    <1> ;     cmp    eax, ecx
6188                                    <1> ;     jnb    short sb16_gcd_3
6189                                    <1> ;     ; remain count < graphics bytes
6190                                    <1> ;     mov    eax, ecx ; fix remain count to data size
6191                                    <1> ;sb16_gcd_3:
6192                                    <1> ;     sub    edi, eax
6193                                    <1> ;     jna    short sb16_gcd_4
6194                                    <1> ;     add    esi, edi ; dma buffer offset
6195                                    <1> ;sb16_gcd_4:
6196                                    <1> ;     mov    edi, ebx ; buffer address (for graphics)
6197                                    <1> ;     mov    [u.r0], ecx
6198                                    <1> ;     rep    movsb
6199                                    <1> ;     retn
6200                                    <1>
6201                                    <1> get_current_sound_data:
6202                                    <1>     ; 24/06/2017
6203                                    <1>     ; 22/06/2017
6204                                    <1>     ; get current sound (PCM out) data for graphics
6205                                    <1>     ;
6206                                    <1>     ; ebx = Physical address (on page boundary)
6207                                    <1>     ; ecx = Byte count
6208                                    <1>     ; [audio_buff_size]
6209                                    <1>
6210                                    <1>     ;mov   edi, [audio_buff_size]
6211 000150F4 8B3D[FC8F0100]            <1>     mov    edi, [audio_dmabuff_size]
6212 000150FA 8B35[F88F0100]            <1>     mov    esi, [audio_dma_buff]
6213 00015100 803D[D98F0100]02          <1>     cmp    byte [audio_device], 2
6214 00015107 72C0                      <1>     jb     short sb16_current_sound_data ; = 1
6215 00015109 D1EF                      <1>     shr    edi, 1
6216 0001510B 39CF                      <1>     cmp    edi, ecx
6217 0001510D 7302                      <1>     jnb    short gcd_0
6218 0001510F 89F9                      <1>     mov    ecx, edi
6219                                    <1> gcd_0:
6220 00015111 803D[D98F0100]03          <1>     cmp    byte [audio_device], 3
6221 00015118 7232                      <1>     jb     short ac97_current_sound_data ; = 2
6222                                    <1>     ; = 3
6223                                    <1> vt8233_current_sound_data:
6224                                    <1>     ; 22/06/2017
6225                                    <1>     ; 21/06/2017
6226                                    <1>     ; get current sound (PCM out) data for graphics
6227                                    <1>     ; (for VT 8233, VT 8237R)
6228                                    <1>     ; ebx = Physical address (on page boundary)
6229                                    <1>     ; ecx = Byte count
6230                                    <1>     ; [audio_buff_size]
6231                                    <1>
6232                                    <1>     ;;mov  edi, [audio_buff_size]
6233                                    <1>     ;mov   edi, [audio_dmabuff_size]
6234                                    <1>     ;mov   esi, [audio_dma_buff]
```

```
                            <1>       ;shr   edi, 1
                            <1>       ;cmp   edi, ecx
                            <1>       ;jnb   short vt8233_gcd_1
                            <1>       ;mov   ecx, edi
                            <1> vt8233_gcd_1:
0001511A BA0C000000         <1>       mov    edx, VIA_REG_OFFSET_CURR_COUNT
0001511F E88FF5FFFF         <1>       call   ctrl_io_r32
00015124 89C2               <1>       mov    edx, eax ; remain count (bits 23-0),
                            <1>                       ; SGD index (bits 31-24)
00015126 81E200000001       <1>       and    edx, 1000000h ; SGD index (0 = 1st half)
0001512C 7402               <1>       jz     short vt8233_gcd_2
                            <1>       ; the second half of DMA buffer
0001512E 01FE               <1>       add    esi, edi
                            <1> vt8233_gcd_2:
00015130 25FFFFFF00         <1>       and    eax, 0FFFFFFh ; bits 23-0
                            <1> ac97_gcd_2:
                            <1> sb16_gcd_2:
00015135 39C8               <1>       cmp    eax, ecx
00015137 7302               <1>       jnb    short vt8233_gcd_3
                            <1>       ; remain count < graphics bytes
00015139 89C8               <1>       mov    eax, ecx ; fix remain count to data size
                            <1> vt8233_gcd_3:
0001513B 29C7               <1>       sub    edi, eax
0001513D 7602               <1>       jna    short vt8233_gcd_4
0001513F 01FE               <1>       add    esi, edi ; dma buffer offset
                            <1> vt8233_gcd_4:
00015141 89DF               <1>       mov    edi, ebx ; buffer address (for graphics)
00015143 890D[64030300]     <1>       mov    [u.r0], ecx
00015149 F3A4               <1>       rep    movsb
                            <1> vt8233_gcd_5:
0001514B C3                 <1>       retn
                            <1>
                            <1> ac97_current_sound_data:
                            <1>       ; 23/06/2017
                            <1>       ; 22/06/2017
                            <1>       ; get current sound (PCM out) data for graphics
                            <1>       ; (for AC'97, ICH)
                            <1>       ; ebx = Physical address (on page boundary)
                            <1>       ; ecx = Byte count
                            <1>       ; [audio_buff_size]
                            <1>
                            <1>       ;;mov  edi, [audio_buff_size]
                            <1>       ;mov   edi, [audio_dmabuff_size]
                            <1>       ;mov   esi, [audio_dma_buff]
                            <1>       ;shr   edi, 1
                            <1>       ;cmp   edi, ecx
                            <1>       ;jnb   short ac97_gcd_0
                            <1>       ;mov   ecx, edi
                            <1> ac97_gcd_0:
0001514C 66BA1400           <1>       mov    dx, PO_CIV_REG ; Position In Current Buff Reg
00015150 660315[12900100]   <1>       add    dx, [NABMBAR]
00015157 EC                 <1>       in     al, dx ; current index value
00015158 A801               <1>       test   al, 1
0001515A 7402               <1>       jz     short ac97_gcd_1
0001515C 01FE               <1>       add    esi, edi
                            <1> ac97_gcd_1:
0001515E 31C0               <1>       xor    eax, eax
00015160 66BA1800           <1>       mov    dx, PO_PICB_REG ; Position In Current Buff Reg
00015164 660315[12900100]   <1>       add    dx, [NABMBAR]
0001516B 66ED               <1>       in     ax, dx ; remain dwords
0001516D C1E002             <1>       shl    eax, 2 ; remain bytes ; 23/06/2017
00015170 EBC3               <1>       jmp    short ac97_gcd_2
                            <1> ;     cmp    eax, ecx
                            <1> ;     jnb    short ac97_gcd_2
                            <1> ;     ; remain count < graphics bytes
                            <1> ;     mov    eax, ecx ; fix remain count to data size
                            <1> ;ac97_gcd_2:
                            <1> ;     sub    edi, eax
                            <1> ;     jna    short ac97_gcd_3
                            <1> ;     add    esi, edi ; dma buffer offset
                            <1> ;ac97_gcd_3:
                            <1> ;     mov    edi, ebx ; buffer address (for graphics)
                            <1> ;     mov    [u.r0], ecx
                            <1> ;     rep    movsb
                            <1> ;     retn
                            <1>
                            <1> sb16_get_dma_buff_off:
                            <1>       ; 28/10/2017
                            <1>       ; 24/06/2017
                            <1>       ; 22/06/2017
                            <1>       ; get current (PCM OUT DMA buffer) pointer
                            <1>       ; (for Sound Blaster 16)
                            <1>
                            <1>       ;mov   ecx, [audio_dmabuff_size]
                            <1>       ;xor   ebx, ebx
                            <1>       ;shr   ecx, 1
                            <1> sb16_gdmabo_0:
                            <1>       ; 28/10/2017
00015172 803D[08900100]10   <1>       cmp    byte [audio_bps], 16
00015179 750F               <1>       jne    short sb16_gdmabo_1 ; 8 bit DMA channel
                            <1>       ; 16 bit DMA channel
0001517B E4C6               <1>       in     al, 0C6h ; DMA channel 5 count register
0001517D 88C2               <1>       mov    dl, al
0001517F E4C6               <1>       in     al, 0C6h
00015181 88C6               <1>       mov    dh, al
00015183 0FB7C2             <1>       movzx  eax, dx
00015186 D1E0               <1>       shl    eax, 1 ; word count -> byte count
00015188 EB3D               <1>       jmp    short sb16_gdmabo_2
                            <1> sb16_gdmabo_1:
0001518A E403               <1>       in     al, 03h ; DMA channel 1 count register
0001518C 88C2               <1>       mov    dl, al
0001518E E403               <1>       in     al, 03h
00015190 88C6               <1>       mov    dh, al
00015192 0FB7C2             <1>       movzx  eax, dx
00015195 EB30               <1>       jmp    short sb16_gdmabo_2
```

```
6340                                    <1>
6341                                    <1> get_dma_buffer_offset:
6342                                    <1>         ; 24/06/2017
6343                                    <1>         ; 22/06/2017
6344                                    <1>         ; get current sound (PCM out) data for graphics
6345                                    <1>         ;
6346                                    <1>         ; ebx = Physical address (on page boundary)
6347                                    <1>         ; ecx = Byte count
6348                                    <1>         ; [audio_buff_size]
6349                                    <1>
6350 00015197 8B0D[FC8F0100]           <1>         mov    ecx, [audio_dmabuff_size]
6351 0001519D 31DB                     <1>         xor    ebx, ebx
6352                                    <1> gdmabo_0:
6353 0001519F 803D[D98F0100]02         <1>         cmp    byte [audio_device], 2
6354 000151A6 72CA                     <1>         jb     short sb16_get_dma_buff_off
6355 000151A8 742A                     <1>         je     short ac97_get_dma_buff_off
6356                                    <1>
6357                                    <1> vt8233_get_dma_buff_off:
6358                                    <1>         ; 24/06/2017
6359                                    <1>         ; 22/06/2017
6360                                    <1>         ; get current (PCM OUT DMA buffer) pointer
6361                                    <1>         ; (for VT 8233, VT 8237R)
6362                                    <1>
6363                                    <1>         ;mov   ecx, [audio_dmabuff_size]
6364                                    <1>         ;xor   ebx, ebx
6365 000151AA D1E9                     <1>         shr    ecx, 1
6366                                    <1> vt8233_gdmabo_0:
6367 000151AC BA0C000000               <1>         mov    edx, VIA_REG_OFFSET_CURR_COUNT
6368 000151B1 E8FDF4FFFF               <1>         call   ctrl_io_r32
6369 000151B6 89C2                     <1>         mov    edx, eax ; remain count (bits 23-0),
6370                                    <1>                         ; SGD index (bits 31-24)
6371 000151B8 81E200000001             <1>         and    edx, 1000000h ; SGD index (0 = 1st half)
6372 000151BE 7402                     <1>         jz     short vt8233_gdmabo_1
6373                                    <1>         ; the second half of DMA buffer
6374 000151C0 89CB                     <1>         mov    ebx, ecx
6375                                    <1> vt8233_gdmabo_1:
6376 000151C2 25FFFFFF00               <1>         and    eax, 0FFFFFFh ; bits 23-0
6377                                    <1> sb16_gdmabo_2:
6378                                    <1> ac97_gdmabo_2:
6379 000151C7 29C1                     <1>         sub    ecx, eax
6380 000151C9 7602                     <1>         jna    short vt8233_gdmabo_2
6381 000151CB 01CB                     <1>         add    ebx, ecx ; dma buffer offset
6382                                    <1> vt8233_gdmabo_2:
6383 000151CD 891D[64030300]           <1>         mov    [u.r0], ebx
6384 000151D3 C3                       <1>         retn
6385                                    <1>
6386                                    <1> ac97_get_dma_buff_off:
6387                                    <1>         ; 24/06/2017
6388                                    <1>         ; 22/06/2017
6389                                    <1>         ; get current (PCM OUT DMA buffer) pointer
6390                                    <1>         ; (for AC'97, ICH)
6391                                    <1>         ; ebx = Physical address (on page boundary)
6392                                    <1>         ; ecx = Byte count
6393                                    <1>         ; [audio_buff_size]
6394                                    <1>
6395                                    <1>         ;mov   ecx, [audio_dmabuff_size]
6396                                    <1>         ;xor   ebx, ebx
6397 000151D4 D1E9                     <1>         shr    ecx, 1
6398                                    <1> ac97_gdmabo_0:
6399 000151D6 66BA1400                 <1>         mov    dx, PO_CIV_REG ; Position In Current Buff Reg
6400 000151DA 660315[12900100]         <1>         add    dx, [NABMBAR]
6401 000151E1 EC                       <1>         in     al, dx ; current index value
6402 000151E2 A801                     <1>         test   al, 1
6403 000151E4 7402                     <1>         jz     short ac97_gdmabo_1
6404 000151E6 89CB                     <1>         mov    ebx, ecx
6405                                    <1> ac97_gdmabo_1:
6406 000151E8 31C0                     <1>         xor    eax, eax
6407 000151EA 66BA1800                 <1>         mov    dx, PO_PICB_REG ; Position In Current Buff Reg
6408 000151EE 660315[12900100]         <1>         add    dx, [NABMBAR]
6409 000151F5 66ED                     <1>         in     ax, dx ; remain dwords
6410 000151F7 EBCE                     <1>         jmp    short ac97_gdmabo_2
3452
3453 000151F9 90<rep 3h>                            align 4
3454
3455                                        %include 'vgadata.s' ; 04/07/2016
3456                                    <1> ; ****************************************************************************************************
3457                                    <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.3 - vgadata.s  (palette and font data)
3458                                    <1> ; ----------------------------------------------------------------------------
3459                                    <1> ; Last Update: 01/01/2021
3460                                    <1> ; ----------------------------------------------------------------------------
3461                                    <1> ; Beginning: 16/01/2016
3462                                    <1> ; ----------------------------------------------------------------------------
3463                                    <1> ; Assembler: NASM version 2.15 (trdos386.s)
3464                                    <1> ; ----------------------------------------------------------------------------
3465                                    <1> ; Turkish Rational DOS
3466                                    <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
3467                                    <1> ;
3468                                    <1> ; Derived from 'Plex86/Bochs VGABios' source code, vgabios-0.7a (2011)
3469                                    <1> ; by the LGPL VGABios Developers Team (2001-2008), 'vgatables.h'
3470                                    <1> ;
3471                                    <1> ; Oracle VirtualBox 5.0.24 VGABios Source Code
3472                                    <1> ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
3473                                    <1> ;
3474                                    <1> ; Palette and font data in assembly language format:
3475                                    <1> ; 'VBoxVgaBiosAlternative.asm'
3476                                    <1>
3477                                    <1> ; ****************************************************************************************************
3478                                    <1>
3479                                    <1> ; 25/11/2020 (TRDOS 386 v2.0.3)
3480                                    <1> ; ('vgatables.h' - 30/12/2019 - vruppert)
3481                                    <1>
3482                                    <1> ; 04/07/2016
3483                                    <1> ; COLOR DATA
3484                                    <1>
3485                                    <1> palette0: ; (63+1)*3
3486 000151FC 000000000000000000-      <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
```

```
3486 00015205 00000000000000      <1>
3487 0001520C 00000000000000002A- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
3487 00015215 2A2A2A2A2A2A2A      <1>
3488 0001521C 2A2A2A2A2A2A2A2A2A- <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
3488 00015225 2A2A2A2A2A2A2A      <1>
3489 0001522C 2A2A2A2A2A2A2A2A2A- <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
3489 00015235 2A2A2A2A2A2A2A      <1>
3490 0001523C 2A2A2A2A2A2A2A2A3F- <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh
3490 00015245 3F3F3F3F3F3F3F      <1>
3491 0001524C 3F3F3F3F3F3F3F3F3F- <1>     db  03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh
3491 00015255 3F3F3F3F3F3F3F      <1>
3492 0001525C 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3492 00015265 00000000000000      <1>
3493 0001526C 00000000000000002A- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
3493 00015275 2A2A2A2A2A2A2A      <1>
3494 0001527C 2A2A2A2A2A2A2A2A2A- <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
3494 00015285 2A2A2A2A2A2A2A      <1>
3495 0001528C 2A2A2A2A2A2A2A2A2A- <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
3495 00015295 2A2A2A2A2A2A2A      <1>
3496 0001529C 2A2A2A2A2A2A2A2A3F- <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh
3496 000152A5 3F3F3F3F3F3F3F      <1>
3497 000152AC 3F3F3F3F3F3F3F3F3F- <1>     db  03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh
3497 000152B5 3F3F3F3F3F3F3F      <1>
3498                              <1> palette1: ; (63+1)*3
3499 000152BC 00000000002A002A00- <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah
3499 000152C5 002A2A2A00002A      <1>
3500 000152CC 002A2A15002A2A2A00- <1>     db  000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah
3500 000152D5 000000002A002A      <1>
3501 000152DC 00002A2A2A00002A00- <1>     db  000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah, 000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah
3501 000152E5 2A2A15002A2A2A      <1>
3502 000152EC 15151515153F153F15- <1>     db  015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh, 015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh
3502 000152F5 153F3F3F15153F      <1>
3503 000152FC 153F3F3F153F3F3F15- <1>     db  015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh
3503 00015305 151515153F153F      <1>
3504 0001530C 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh
3504 00015315 3F3F3F153F3F3F      <1>
3505 0001531C 00000000002A002A00- <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah
3505 00015325 002A2A2A00002A      <1>
3506 0001532C 002A2A15002A2A2A00- <1>     db  000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah
3506 00015335 000000002A002A      <1>
3507 0001533C 00002A2A2A00002A00- <1>     db  000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah, 000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah
3507 00015345 2A2A15002A2A2A      <1>
3508 0001534C 15151515153F153F15- <1>     db  015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh, 015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh
3508 00015355 153F3F3F15153F      <1>
3509 0001535C 153F3F3F153F3F3F15- <1>     db  015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh
3509 00015365 151515153F153F      <1>
3510 0001536C 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh
3510 00015375 3F3F3F153F3F3F      <1>
3511                              <1> palette2: ; (63+1)*3
3512 0001537C 00000000002A002A00- <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah
3512 00015385 002A2A2A00002A      <1>
3513 0001538C 002A2A2A002A2A2A00- <1>     db  000h, 02ah, 02ah, 02ah, 000h, 02ah, 02ah, 02ah, 000h, 000h, 015h, 000h, 000h, 03fh, 000h, 02ah
3513 00015395 001500003F002A      <1>
3514 0001539C 15002A3F2A00152A00- <1>     db  015h, 000h, 02ah, 03fh, 02ah, 000h, 015h, 02ah, 000h, 03fh, 02ah, 02ah, 015h, 02ah, 02ah, 03fh
3514 000153A5 3F2A2A152A2A3F      <1>
3515 000153AC 00150000152A003F00- <1>     db  000h, 015h, 000h, 000h, 015h, 02ah, 000h, 03fh, 000h, 000h, 03fh, 02ah, 02ah, 015h, 000h, 02ah
3515 000153B5 003F2A2A15002A      <1>
3516 000153BC 152A2A3F002A3F2A00- <1>     db  015h, 02ah, 02ah, 03fh, 000h, 02ah, 03fh, 02ah, 000h, 015h, 015h, 000h, 015h, 03fh, 000h, 03fh
3516 000153C5 151500153F003F      <1>
3517 000153CC 15003F3F2A15152A15- <1>     db  015h, 000h, 03fh, 03fh, 02ah, 015h, 015h, 02ah, 015h, 03fh, 02ah, 03fh, 015h, 02ah, 03fh, 03fh
3517 000153D5 3F2A3F152A3F3F      <1>
3518 000153DC 15000015002A152A00- <1>     db  015h, 000h, 000h, 015h, 000h, 02ah, 015h, 02ah, 000h, 015h, 02ah, 02ah, 03fh, 000h, 000h, 03fh
3518 000153E5 152A2A3F00003F      <1>
3519 000153EC 002A3F2A003F2A2A15- <1>     db  000h, 02ah, 03fh, 02ah, 000h, 03fh, 02ah, 02ah, 015h, 000h, 015h, 015h, 000h, 03fh, 015h, 02ah
3519 000153F5 001515003F152A      <1>
3520 000153FC 15152A3F3F00153F00- <1>     db  015h, 015h, 02ah, 03fh, 03fh, 000h, 015h, 03fh, 000h, 03fh, 03fh, 02ah, 015h, 03fh, 02ah, 03fh
3520 00015405 3F3F2A153F2A3F      <1>
3521 0001540C 15150015152A153F00- <1>     db  015h, 015h, 000h, 015h, 015h, 02ah, 015h, 03fh, 000h, 015h, 03fh, 02ah, 03fh, 015h, 000h, 03fh
3521 00015415 153F2A3F15003F      <1>
3522 0001541C 152A3F3F003F3F2A15- <1>     db  015h, 02ah, 03fh, 03fh, 000h, 03fh, 03fh, 02ah, 015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh
3522 00015425 151515153F153F      <1>
3523 0001542C 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh
3523 00015435 3F3F3F153F3F3F      <1>
3524                              <1> palette3: ; 256*3
3525 0001543C 00000000002A002A00- <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah
3525 00015445 002A2A2A00002A      <1>
3526 0001544C 002A2A15002A2A2A15- <1>     db  000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah, 015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh
3526 00015455 151515153F153F      <1>
3527 0001545C 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh
3527 00015465 3F3F3F153F3F3F      <1>
3528 0001546C 000000050505080808- <1>     db  000h, 000h, 000h, 005h, 005h, 005h, 008h, 008h, 008h, 00bh, 00bh, 00bh, 00eh, 00eh, 00eh, 011h
3528 00015475 0B0B0B0E0E0E11      <1>
3529 0001547C 11111414141818181C- <1>     db  011h, 011h, 014h, 014h, 014h, 018h, 018h, 018h, 01ch, 01ch, 01ch, 020h, 020h, 020h, 024h, 024h
3529 00015485 1C1C2020202424      <1>
3530 0001548C 242828282D2D2D3232- <1>     db  024h, 028h, 028h, 028h, 02dh, 02dh, 02dh, 032h, 032h, 032h, 038h, 038h, 038h, 03fh, 03fh, 03fh
3530 00015495 323838383F3F3F      <1>
3531 0001549C 00003F10003F1F003F- <1>     db  000h, 000h, 03fh, 010h, 000h, 03fh, 01fh, 000h, 03fh, 02fh, 000h, 03fh, 03fh, 000h, 03fh, 03fh
3531 000154A5 2F003F3F003F3F      <1>
3532 000154AC 002F3F001F3F00103F- <1>     db  000h, 02fh, 03fh, 000h, 01fh, 03fh, 000h, 010h, 03fh, 000h, 000h, 03fh, 010h, 000h, 03fh, 01fh
3532 000154B5 00003F10003F1F      <1>
3533 000154BC 003F2F003F3F002F3F- <1>     db  000h, 03fh, 02fh, 000h, 03fh, 03fh, 000h, 02fh, 03fh, 000h, 01fh, 03fh, 000h, 010h, 03fh, 000h
3533 000154C5 001F3F00103F00      <1>
3534 000154CC 003F00003F10003F1F- <1>     db  000h, 03fh, 000h, 000h, 03fh, 010h, 000h, 03fh, 01fh, 000h, 03fh, 02fh, 000h, 03fh, 03fh, 000h
3534 000154D5 003F2F003F3F00      <1>
3535 000154DC 2F3F001F3F00103F1F- <1>     db  02fh, 03fh, 000h, 01fh, 03fh, 000h, 010h, 03fh, 01fh, 01fh, 03fh, 027h, 01fh, 03fh, 02fh, 01fh
3535 000154E5 1F3F271F3F2F1F      <1>
3536 000154EC 3F371F3F3F1F3F3F1F- <1>     db  03fh, 037h, 01fh, 03fh, 03fh, 01fh, 03fh, 03fh, 01fh, 037h, 03fh, 01fh, 02fh, 03fh, 01fh, 027h
3536 000154F5 373F1F2F3F1F27      <1>
3537 000154FC 3F1F1F3F271F3F2F1F- <1>     db  03fh, 01fh, 01fh, 03fh, 027h, 01fh, 03fh, 02fh, 01fh, 03fh, 037h, 01fh, 03fh, 03fh, 01fh, 037h
3537 00015505 3F371F3F3F1F37      <1>
3538 0001550C 3F1F2F3F1F273F1F1F- <1>     db  03fh, 01fh, 02fh, 03fh, 01fh, 027h, 03fh, 01fh, 01fh, 03fh, 01fh, 01fh, 03fh, 027h, 01fh, 03fh
3538 00015515 3F1F1F3F271F3F      <1>
3539 0001551C 2F1F3F371F3F3F1F37- <1>     db  02fh, 01fh, 03fh, 037h, 01fh, 03fh, 03fh, 01fh, 037h, 03fh, 01fh, 02fh, 03fh, 01fh, 027h, 03fh
3539 00015525 3F1F2F3F1F273F      <1>
3540 0001552C 2D2D3F312D3F362D3F- <1>     db  02dh, 02dh, 03fh, 031h, 02dh, 03fh, 036h, 02dh, 03fh, 03ah, 02dh, 03fh, 03fh, 02dh, 03fh, 03fh
3540 00015535 3A2D3F3F2D3F3F      <1>
3541 0001553C 2D3A3F2D363F2D313F- <1>     db  02dh, 03ah, 03fh, 02dh, 036h, 03fh, 02dh, 031h, 03fh, 02dh, 02dh, 03fh, 031h, 02dh, 03fh, 036h
3541 00015545 2D2D3F312D3F36      <1>
3542 0001554C 2D3F3A2D3F3F2D3A3F- <1>     db  02dh, 03fh, 03ah, 02dh, 03fh, 03fh, 02dh, 03ah, 03fh, 02dh, 036h, 03fh, 02dh, 031h, 03fh, 02dh
3542 00015555 2D363F2D313F2D      <1>
3543 0001555C 2D3F2D2D3F312D3F36- <1>     db  02dh, 03fh, 02dh, 02dh, 03fh, 031h, 02dh, 03fh, 036h, 02dh, 03fh, 03ah, 02dh, 03fh, 03fh, 02dh
```

```
3543 00015565 2D3F3A2D3F3F2D      <1>
3544 0001556C 3A3F2D363F2D313F00- <1>     db  03ah, 03fh, 02dh, 036h, 03fh, 02dh, 031h, 03fh, 000h, 000h, 01ch, 007h, 000h, 01ch, 00eh, 000h
3544 00015575 001C07001C0E00      <1>
3545 0001557C 1C15001C1C001C1C00- <1>     db  01ch, 015h, 000h, 01ch, 01ch, 000h, 01ch, 01ch, 000h, 015h, 01ch, 000h, 00eh, 01ch, 000h, 007h
3545 00015585 151C000E1C0007      <1>
3546 0001558C 1C00001C07001C0E00- <1>     db  01ch, 000h, 000h, 01ch, 007h, 000h, 01ch, 00eh, 000h, 01ch, 015h, 000h, 01ch, 01ch, 000h, 015h
3546 00015595 1C15001C1C0015      <1>
3547 0001559C 1C000E1C00071C0000- <1>     db  01ch, 000h, 00eh, 01ch, 000h, 007h, 01ch, 000h, 000h, 01ch, 000h, 000h, 01ch, 007h, 000h, 01ch
3547 000155A5 1C00001C07001C      <1>
3548 000155AC 0E001C15001C1C0015- <1>     db  00eh, 000h, 01ch, 015h, 000h, 01ch, 01ch, 000h, 015h, 01ch, 000h, 00eh, 01ch, 000h, 007h, 01ch
3548 000155B5 1C000E1C00071C      <1>
3549 000155BC 0E0E1C110E1C150E1C- <1>     db  00eh, 00eh, 01ch, 011h, 00eh, 01ch, 015h, 00eh, 01ch, 018h, 00eh, 01ch, 01ch, 00eh, 01ch, 01ch
3549 000155C5 180E1C1C0E1C1C      <1>
3550 000155CC 0E181C0E151C0E111C- <1>     db  00eh, 018h, 01ch, 00eh, 015h, 01ch, 00eh, 011h, 01ch, 00eh, 00eh, 01ch, 011h, 00eh, 01ch, 015h
3550 000155D5 0E0E1C110E1C15      <1>
3551 000155DC 0E1C180E1C1C0E181C- <1>     db  00eh, 01ch, 018h, 00eh, 01ch, 01ch, 00eh, 018h, 01ch, 00eh, 015h, 01ch, 00eh, 011h, 01ch, 00eh
3551 000155E5 0E151C0E111C0E      <1>
3552 000155EC 0E1C0E0E1C110E1C15- <1>     db  00eh, 01ch, 00eh, 00eh, 01ch, 011h, 00eh, 01ch, 015h, 00eh, 01ch, 018h, 00eh, 01ch, 01ch, 00eh
3552 000155F5 0E1C180E1C1C0E      <1>
3553 000155FC 181C0E151C0E111C14- <1>     db  018h, 01ch, 00eh, 015h, 01ch, 00eh, 011h, 01ch, 014h, 014h, 01ch, 016h, 014h, 01ch, 018h, 014h
3553 00015605 141C16141C1814      <1>
3554 0001560C 1C1A141C1C141C1C14- <1>     db  01ch, 01ah, 014h, 01ch, 01ch, 014h, 01ch, 01ch, 014h, 01ah, 01ch, 014h, 018h, 01ch, 014h, 016h
3554 00015615 1A1C14181C1416      <1>
3555 0001561C 1C14141C16141C1814- <1>     db  01ch, 014h, 014h, 01ch, 016h, 014h, 01ch, 018h, 014h, 01ch, 01ah, 014h, 01ch, 01ch, 014h, 01ah
3555 00015625 1C1A141C1C141A      <1>
3556 0001562C 1C14181C14161C1414- <1>     db  01ch, 014h, 018h, 01ch, 014h, 016h, 01ch, 014h, 014h, 01ch, 014h, 014h, 01ch, 016h, 014h, 01ch
3556 00015635 1C14141C16141C      <1>
3557 0001563C 18141C1A141C1C141A- <1>     db  018h, 014h, 01ch, 01ah, 014h, 01ch, 01ch, 014h, 01ah, 01ch, 014h, 018h, 01ch, 014h, 016h, 01ch
3557 00015645 1C14181C14161C      <1>
3558 0001564C 0000100400100800 10- <1>     db  000h, 000h, 010h, 004h, 000h, 010h, 008h, 000h, 010h, 00ch, 000h, 010h, 010h, 000h, 010h, 010h
3558 00015655 0C001010001010      <1>
3559 0001565C 000C10000810000410- <1>     db  000h, 00ch, 010h, 000h, 008h, 010h, 000h, 004h, 010h, 000h, 000h, 010h, 004h, 000h, 010h, 008h
3559 00015665 00001004001008      <1>
3560 0001566C 00100C001010000C10- <1>     db  000h, 010h, 00ch, 000h, 010h, 010h, 000h, 00ch, 010h, 000h, 008h, 010h, 000h, 004h, 010h, 000h
3560 00015675 00081000041000      <1>
3561 0001567C 0010000010040010 08- <1>     db  000h, 010h, 000h, 000h, 010h, 004h, 000h, 010h, 008h, 000h, 010h, 00ch, 000h, 010h, 010h, 000h
3561 00015685 00100C00101000      <1>
3562 0001568C 0C1000081000041008- <1>     db  00ch, 010h, 000h, 008h, 010h, 000h, 004h, 010h, 008h, 008h, 010h, 00ah, 008h, 010h, 00ch, 008h
3562 00015695 08100A08100C08      <1>
3563 0001569C 100E08101008101008- <1>     db  010h, 00eh, 008h, 010h, 010h, 008h, 010h, 010h, 008h, 00eh, 010h, 008h, 00ch, 010h, 008h, 00ah
3563 000156A5 0E10080C10080A      <1>
3564 000156AC 100808100A08100C08- <1>     db  010h, 008h, 008h, 010h, 00ah, 008h, 010h, 00ch, 008h, 010h, 00eh, 008h, 010h, 010h, 008h, 00eh
3564 000156B5 100E081010080E      <1>
3565 000156BC 10080C10080A100808- <1>     db  010h, 008h, 00ch, 010h, 008h, 00ah, 010h, 008h, 008h, 010h, 008h, 008h, 010h, 00ah, 008h, 010h
3565 000156C5 100808100A0810      <1>
3566 000156CC 0C08100E081010080E- <1>     db  00ch, 008h, 010h, 00eh, 008h, 010h, 010h, 008h, 00eh, 010h, 008h, 00ch, 010h, 008h, 00ah, 010h
3566 000156D5 10080C10080A10      <1>
3567 000156DC 0B0B100C0B100D0B10- <1>     db  00bh, 00bh, 010h, 00ch, 00bh, 010h, 00dh, 00bh, 010h, 00fh, 00bh, 010h, 010h, 00bh, 010h, 010h
3567 000156E5 0F0B10100B1010      <1>
3568 000156EC 0B0F100B0D100B0C10- <1>     db  00bh, 00fh, 010h, 00bh, 00dh, 010h, 00bh, 00ch, 010h, 00bh, 00bh, 010h, 00ch, 00bh, 010h, 00dh
3568 000156F5 0B0B100C0B100D      <1>
3569 000156FC 0B100F0B10100B0F10- <1>     db  00bh, 010h, 00fh, 00bh, 010h, 010h, 00bh, 00fh, 010h, 00bh, 00dh, 010h, 00bh, 00ch, 010h, 00bh
3569 00015705 0B0D100B0C100B      <1>
3570 0001570C 0B100B0B100C0B100D- <1>     db  00bh, 010h, 00bh, 00bh, 010h, 00ch, 00bh, 010h, 00dh, 00bh, 010h, 00fh, 00bh, 010h, 010h, 00bh
3570 00015715 0B100F0B10100B      <1>
3571 0001571C 0F100B0D100B0C1000- <1>     db  00fh, 010h, 00bh, 00dh, 010h, 00bh, 00ch, 010h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3571 00015725 00000000000000      <1>
3572 0001572C 0000000000000000 00- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3572 00015735 00000000000000      <1>
3573                              <1>
3574                              <1>
3575                              <1> ; 04/07/2016
3576                              <1> ; FONT DATA
3577                              <1>
3578                              <1> CRT_CHAR_GEN:
3579                              <1> vgafont8:
3580 0001573C 0000000000000000 7E- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 081h, 0a5h, 081h, 0bdh, 099h, 081h, 07eh
3580 00015745 81A581BD99817E      <1>
3581 0001574C 7EFFDBFFC3E7FF7E6C- <1>     db  07eh, 0ffh, 0dbh, 0ffh, 0c3h, 0e7h, 0ffh, 07eh, 06ch, 0feh, 0feh, 0feh, 07ch, 038h, 010h, 000h
3581 00015755 FEFEFE7C381000      <1>
3582 0001575C 10387CFE7C38100038- <1>     db  010h, 038h, 07ch, 0feh, 07ch, 038h, 010h, 000h, 038h, 07ch, 038h, 0feh, 0feh, 07ch, 038h, 07ch
3582 00015765 7C38FEFE7C387C      <1>
3583 0001576C 1010387CFE7C387C00- <1>     db  010h, 010h, 038h, 07ch, 0feh, 07ch, 038h, 07ch, 000h, 000h, 018h, 03ch, 03ch, 018h, 000h, 000h
3583 00015775 00183C3C180000      <1>
3584 0001577C FFFFE7C3C3E7FFFF00- <1>     db  0ffh, 0ffh, 0e7h, 0c3h, 0c3h, 0e7h, 0ffh, 0ffh, 000h, 03ch, 066h, 042h, 042h, 066h, 03ch, 000h
3584 00015785 3C664242663C00      <1>
3585 0001578C FFC399BDBD99C3FF0F- <1>     db  0ffh, 0c3h, 099h, 0bdh, 0bdh, 099h, 0c3h, 0ffh, 00fh, 007h, 00fh, 07dh, 0cch, 0cch, 0cch, 078h
3585 00015795 070F7DCCCCCC78      <1>
3586 0001579C 3C6666663C187E183F- <1>     db  03ch, 066h, 066h, 066h, 03ch, 018h, 07eh, 018h, 03fh, 033h, 03fh, 030h, 030h, 070h, 0f0h, 0e0h
3586 000157A5 333F303070F0E0      <1>
3587 000157AC 7F637F636367E6C099- <1>     db  07fh, 063h, 07fh, 063h, 063h, 067h, 0e6h, 0c0h, 099h, 05ah, 03ch, 0e7h, 0e7h, 03ch, 05ah, 099h
3587 000157B5 5A3CE7E73C5A99      <1>
3588 000157BC 80E0F8FEF8E0800002- <1>     db  080h, 0e0h, 0f8h, 0feh, 0f8h, 0e0h, 080h, 000h, 002h, 00eh, 03eh, 0feh, 03eh, 00eh, 002h, 000h
3588 000157C5 0E3EFE3E0E0200      <1>
3589 000157CC 183C7E18187E3C1866- <1>     db  018h, 03ch, 07eh, 018h, 018h, 07eh, 03ch, 018h, 066h, 066h, 066h, 066h, 066h, 000h, 066h, 000h
3589 000157D5 66666666006600      <1>
3590 000157DC 7FDBDB7B1B1B1B003E- <1>     db  07fh, 0dbh, 0dbh, 07bh, 01bh, 01bh, 01bh, 000h, 03eh, 063h, 038h, 06ch, 06ch, 038h, 0cch, 078h
3590 000157E5 63386C6C38CC78      <1>
3591 000157EC 000000007E7E7E0018- <1>     db  000h, 000h, 000h, 000h, 07eh, 07eh, 07eh, 000h, 018h, 03ch, 07eh, 018h, 07eh, 03ch, 018h, 0ffh
3591 000157F5 3C7E187E3C18FF      <1>
3592 000157FC 183C7E1818181800 18- <1>     db  018h, 03ch, 07eh, 018h, 018h, 018h, 018h, 000h, 018h, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h
3592 00015805 1818187E3C1800      <1>
3593 0001580C 00180CFE0C18000000- <1>     db  000h, 018h, 00ch, 0feh, 00ch, 018h, 000h, 000h, 000h, 030h, 060h, 0feh, 060h, 030h, 000h, 000h
3593 00015815 3060FE60300000      <1>
3594 0001581C 0000C0C0C0FE000000- <1>     db  000h, 000h, 0c0h, 0c0h, 0c0h, 0feh, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h
3594 00015825 2466FF66240000      <1>
3595 0001582C 00183C7EFFFF000000- <1>     db  000h, 018h, 03ch, 07eh, 0ffh, 0ffh, 000h, 000h, 000h, 0ffh, 0ffh, 07eh, 03ch, 018h, 000h, 000h
3595 00015835 FFFF7E3C180000      <1>
3596 0001583C 0000000000000030 00- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 030h, 078h, 078h, 030h, 030h, 000h, 030h, 000h
3596 00015845 78783030003000      <1>
3597 0001584C 6C6C6C000000000006C- <1>     db  06ch, 06ch, 06ch, 000h, 000h, 000h, 000h, 000h, 06ch, 06ch, 0feh, 06ch, 0feh, 06ch, 06ch, 000h
3597 00015855 6CFE6CFE6C6C00      <1>
3598 0001585C 307CC0780CF8300000- <1>     db  030h, 07ch, 0c0h, 078h, 00ch, 0f8h, 030h, 000h, 000h, 0c6h, 0cch, 018h, 030h, 066h, 0c6h, 000h
3598 00015865 C6CC183066C600      <1>
3599 0001586C 386C3876DCCC760060- <1>     db  038h, 06ch, 038h, 076h, 0dch, 0cch, 076h, 000h, 060h, 060h, 0c0h, 000h, 000h, 000h, 000h, 000h
3599 00015875 60C000000000000     <1>
3600 0001587C 1830606060301800 60- <1>     db  018h, 030h, 060h, 060h, 060h, 030h, 018h, 000h, 060h, 030h, 018h, 018h, 018h, 030h, 060h, 000h
3600 00015885 301818183060 00     <1>
3601 0001588C 00663CFF3C66000000- <1>     db  000h, 066h, 03ch, 0ffh, 03ch, 066h, 000h, 000h, 000h, 030h, 030h, 0fch, 030h, 030h, 000h, 000h
3601 00015895 3030FC30300000      <1>
3602 0001589C 0000000000306000 00- <1>     db  000h, 000h, 000h, 000h, 000h, 030h, 030h, 060h, 000h, 000h, 000h, 0fch, 000h, 000h, 000h, 000h
```

```
3602 000158A5 0000FC00000000      <1>
3603 000158AC 000000000030300006- <1>     db  000h, 000h, 000h, 000h, 000h, 030h, 030h, 000h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 080h, 000h
3603 000158B5 0C183060C08000      <1>
3604 000158BC 7CC6CEDEF6E67C0030- <1>     db  07ch, 0c6h, 0ceh, 0deh, 0f6h, 0e6h, 07ch, 000h, 030h, 070h, 030h, 030h, 030h, 030h, 0fch, 000h
3604 000158C5 7030303030FC00      <1>
3605 000158CC 78CC0C3860CCFC0078- <1>     db  078h, 0cch, 00ch, 038h, 060h, 0cch, 0fch, 000h, 078h, 0cch, 00ch, 038h, 00ch, 0cch, 078h, 000h
3605 000158D5 CC0C380CCC7800      <1>
3606 000158DC 1C3C6CCCFE0C1E00FC- <1>     db  01ch, 03ch, 06ch, 0cch, 0feh, 00ch, 01eh, 000h, 0fch, 0c0h, 0f8h, 00ch, 00ch, 0cch, 078h, 000h
3606 000158E5 C0F80C0CCC7800      <1>
3607 000158EC 3860C0F8CCCC7800FC- <1>     db  038h, 060h, 0c0h, 0f8h, 0cch, 0cch, 078h, 000h, 0fch, 0cch, 00ch, 018h, 030h, 030h, 030h, 000h
3607 000158F5 CC0C1830303000      <1>
3608 000158FC 78CCCC78CCCC780078- <1>     db  078h, 0cch, 0cch, 078h, 0cch, 0cch, 078h, 000h, 078h, 0cch, 0cch, 07ch, 00ch, 018h, 070h, 000h
3608 00015905 CCCC7C0C187000      <1>
3609 0001590C 003030000030300000- <1>     db  000h, 030h, 030h, 000h, 000h, 030h, 030h, 000h, 000h, 030h, 030h, 000h, 000h, 030h, 030h, 060h
3609 00015915 303000000303060     <1>
3610 0001591C 183060C06030180000- <1>     db  018h, 030h, 060h, 0c0h, 060h, 030h, 018h, 000h, 000h, 000h, 0fch, 000h, 000h, 0fch, 000h, 000h
3610 00015925 00FC0000FC0000      <1>
3611 0001592C 6030180C1830600078- <1>     db  060h, 030h, 018h, 00ch, 018h, 030h, 060h, 000h, 078h, 0cch, 00ch, 018h, 030h, 000h, 030h, 000h
3611 00015935 CC0C1830003000      <1>
3612 0001593C 7CC6DEDEDEC0780030- <1>     db  07ch, 0c6h, 0deh, 0deh, 0deh, 0c0h, 078h, 000h, 030h, 078h, 0cch, 0cch, 0fch, 0cch, 0cch, 000h
3612 00015945 78CCCCFCCCCC00      <1>
3613 0001594C FC66667C6666FC003C- <1>     db  0fch, 066h, 066h, 07ch, 066h, 066h, 0fch, 000h, 03ch, 066h, 0c0h, 0c0h, 0c0h, 066h, 03ch, 000h
3613 00015955 66C0C0C0663C00      <1>
3614 0001595C F86C6666666CF800FE- <1>     db  0f8h, 06ch, 066h, 066h, 066h, 06ch, 0f8h, 000h, 0feh, 062h, 068h, 078h, 068h, 062h, 0feh, 000h
3614 00015965 6268786862FE00      <1>
3615 0001596C FE6268786860F0003C- <1>     db  0feh, 062h, 068h, 078h, 068h, 060h, 0f0h, 000h, 03ch, 066h, 0c0h, 0c0h, 0ceh, 066h, 03eh, 000h
3615 00015975 66C0C0CE663E00      <1>
3616 0001597C CCCCCCFCCCCCCC0078- <1>     db  0cch, 0cch, 0cch, 0fch, 0cch, 0cch, 0cch, 000h, 078h, 030h, 030h, 030h, 030h, 030h, 078h, 000h
3616 00015985 30303030307800      <1>
3617 0001598C 1E0C0C0CCCCC7800E6- <1>     db  01eh, 00ch, 00ch, 00ch, 0cch, 0cch, 078h, 000h, 0e6h, 066h, 06ch, 078h, 06ch, 066h, 0e6h, 000h
3617 00015995 666C786C66E600      <1>
3618 0001599C F06060606266FE00C6- <1>     db  0f0h, 060h, 060h, 060h, 062h, 066h, 0feh, 000h, 0c6h, 0eeh, 0feh, 0feh, 0d6h, 0c6h, 0c6h, 000h
3618 000159A5 EEFEFED6C6C600      <1>
3619 000159AC C6E6F6DECEC6C60038- <1>     db  0c6h, 0e6h, 0f6h, 0deh, 0ceh, 0c6h, 0c6h, 000h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 06ch, 038h, 000h
3619 000159B5 6CC6C6C66C3800      <1>
3620 000159BC FC66667C6060F00078- <1>     db  0fch, 066h, 066h, 07ch, 060h, 060h, 0f0h, 000h, 078h, 0cch, 0cch, 0cch, 0dch, 078h, 01ch, 000h
3620 000159C5 CCCCCCDC781C00      <1>
3621 000159CC FC66667C6C66E60078- <1>     db  0fch, 066h, 066h, 07ch, 06ch, 066h, 0e6h, 000h, 078h, 0cch, 0e0h, 070h, 01ch, 0cch, 078h, 000h
3621 000159D5 CCE0701CCC7800      <1>
3622 000159DC FCB4303030307800CC- <1>     db  0fch, 0b4h, 030h, 030h, 030h, 030h, 078h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 0fch, 000h
3622 000159E5 CCCCCCCCCCFC00      <1>
3623 000159EC CCCCCCCCCC783000C6- <1>     db  0cch, 0cch, 0cch, 0cch, 0cch, 078h, 030h, 000h, 0c6h, 0c6h, 0c6h, 0d6h, 0feh, 0eeh, 0c6h, 000h
3623 000159F5 C6C6D6FEEEC600      <1>
3624 000159FC C6C66C38386CC600CC- <1>     db  0c6h, 0c6h, 06ch, 038h, 038h, 06ch, 0c6h, 000h, 0cch, 0cch, 0cch, 078h, 030h, 030h, 078h, 000h
3624 00015A05 CCCC7830307800      <1>
3625 00015A0C FEC68C183266FE0078- <1>     db  0feh, 0c6h, 08ch, 018h, 032h, 066h, 0feh, 000h, 078h, 060h, 060h, 060h, 060h, 060h, 078h, 000h
3625 00015A15 606060606078 00     <1>
3626 00015A1C C06030180C06020078- <1>     db  0c0h, 060h, 030h, 018h, 00ch, 006h, 002h, 000h, 078h, 018h, 018h, 018h, 018h, 018h, 078h, 000h
3626 00015A25 18181818187800      <1>
3627 00015A2C 10386CC60000000000- <1>     db  010h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh
3627 00015A35 000000000000FF      <1>
3628 00015A3C 303018000000000000- <1>     db  030h, 030h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 078h, 00ch, 07ch, 0cch, 076h, 000h
3628 00015A45 00780C7CCC7600      <1>
3629 00015A4C E060607C6666DC0000- <1>     db  0e0h, 060h, 060h, 07ch, 066h, 066h, 0dch, 000h, 000h, 000h, 078h, 0cch, 0c0h, 0cch, 078h, 000h
3629 00015A55 0078CCC0CC7800      <1>
3630 00015A5C 1C0C0C7CCCCC760000- <1>     db  01ch, 00ch, 00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h
3630 00015A65 0078CCFCC07800      <1>
3631 00015A6C 386C60F06060F00000- <1>     db  038h, 06ch, 060h, 0f0h, 060h, 060h, 0f0h, 000h, 000h, 000h, 076h, 0cch, 0cch, 07ch, 00ch, 0f8h
3631 00015A75 0076CCCC7C0CF8      <1>
3632 00015A7C E0606C766666E60030- <1>     db  0e0h, 060h, 06ch, 076h, 066h, 066h, 0e6h, 000h, 030h, 000h, 070h, 030h, 030h, 030h, 078h, 000h
3632 00015A85 00703030307800      <1>
3633 00015A8C 0C000C0C0CCCCC78E0- <1>     db  00ch, 000h, 00ch, 00ch, 00ch, 0cch, 0cch, 078h, 0e0h, 060h, 066h, 06ch, 078h, 06ch, 0e6h, 000h
3633 00015A95 60666C786CE600      <1>
3634 00015A9C 703030303030780000- <1>     db  070h, 030h, 030h, 030h, 030h, 030h, 078h, 000h, 000h, 000h, 0cch, 0feh, 0feh, 0d6h, 0c6h, 000h
3634 00015AA5 00CCFEFED6C600      <1>
3635 00015AAC 0000F8CCCCCCCC0000- <1>     db  000h, 000h, 0f8h, 0cch, 0cch, 0cch, 0cch, 000h, 000h, 000h, 078h, 0cch, 0cch, 0cch, 078h, 000h
3635 00015AB5 0078CCCCCC7800      <1>
3636 00015ABC 0000DC66667C60F000- <1>     db  000h, 000h, 0dch, 066h, 066h, 07ch, 060h, 0f0h, 000h, 000h, 076h, 0cch, 0cch, 07ch, 00ch, 01eh
3636 00015AC5 0076CCCC7C0C1E      <1>
3637 00015ACC 0000DC766660F00000- <1>     db  000h, 000h, 0dch, 076h, 066h, 060h, 0f0h, 000h, 000h, 000h, 07ch, 0c0h, 078h, 00ch, 0f8h, 000h
3637 00015AD5 007CC0780CF800      <1>
3638 00015ADC 10307C303034180000- <1>     db  010h, 030h, 07ch, 030h, 030h, 034h, 018h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 076h, 000h
3638 00015AE5 00CCCCCCCC7600      <1>
3639 00015AEC 0000CCCCCC78300000- <1>     db  000h, 000h, 0cch, 0cch, 0cch, 078h, 030h, 000h, 000h, 000h, 0c6h, 0d6h, 0feh, 0feh, 06ch, 000h
3639 00015AF5 00C6D6FEFE6C00      <1>
3640 00015AFC 0000C66C386CC60000- <1>     db  000h, 000h, 0c6h, 06ch, 038h, 06ch, 0c6h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 07ch, 00ch, 0f8h
3640 00015B05 00CCCCCC7C0CF8      <1>
3641 00015B0C 0000FC983064FC001C- <1>     db  000h, 000h, 0fch, 098h, 030h, 064h, 0fch, 000h, 01ch, 030h, 030h, 0e0h, 030h, 030h, 01ch, 000h
3641 00015B15 3030E030301C00      <1>
3642 00015B1C 18181800181818800E0- <1>     db  018h, 018h, 018h, 000h, 018h, 018h, 018h, 000h, 0e0h, 030h, 030h, 01ch, 030h, 030h, 0e0h, 000h
3642 00015B25 30301C3030E000      <1>
3643 00015B2C 76DC000000000000000- <1>     db  076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 000h
3643 00015B35 10386CC6C6FE00      <1>
3644 00015B3C 78CCC0CC78180C7800- <1>     db  078h, 0cch, 0c0h, 0cch, 078h, 018h, 00ch, 078h, 000h, 0cch, 000h, 0cch, 0cch, 0cch, 07eh, 000h
3644 00015B45 CC00CCCCCC7E00      <1>
3645 00015B4C 1C0078CCFCC078007E- <1>     db  01ch, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h, 07eh, 0c3h, 03ch, 006h, 03eh, 066h, 03fh, 000h
3645 00015B55 C33C063E663F00      <1>
3646 00015B5C CC00780C7CCC7E00E0- <1>     db  0cch, 000h, 078h, 00ch, 07ch, 0cch, 07eh, 000h, 0e0h, 000h, 078h, 00ch, 07ch, 0cch, 07eh, 000h
3646 00015B65 00780C7CCC7E00      <1>
3647 00015B6C 3030780C7CCC7E0000- <1>     db  030h, 030h, 078h, 00ch, 07ch, 0cch, 07eh, 000h, 000h, 000h, 078h, 0c0h, 0c0h, 078h, 00ch, 038h
3647 00015B75 0078C0C0780C38      <1>
3648 00015B7C 7EC33C667E603C00CC- <1>     db  07eh, 0c3h, 03ch, 066h, 07eh, 060h, 03ch, 000h, 0cch, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h
3648 00015B85 0078CCFCC07800      <1>
3649 00015B8C E00078CCFCC07800CC- <1>     db  0e0h, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h, 0cch, 000h, 070h, 030h, 030h, 030h, 078h, 000h
3649 00015B95 00703030307800      <1>
3650 00015B9C 7CC6381818183C00E0- <1>     db  07ch, 0c6h, 038h, 018h, 018h, 018h, 03ch, 000h, 0e0h, 000h, 070h, 030h, 030h, 030h, 078h, 000h
3650 00015BA5 00703030307800      <1>
3651 00015BAC C6386CC6FEC6C60030- <1>     db  0c6h, 038h, 06ch, 0c6h, 0feh, 0c6h, 0c6h, 000h, 030h, 030h, 000h, 078h, 0cch, 0fch, 0cch, 000h
3651 00015BB5 300078CCFCCC00      <1>
3652 00015BBC 1C00FC607860FC0000- <1>     db  01ch, 000h, 0fch, 060h, 078h, 060h, 0fch, 000h, 000h, 000h, 07fh, 00ch, 07fh, 0cch, 07fh, 000h
3652 00015BC5 007F0C7FCC7F00      <1>
3653 00015BCC 3E6CCCFECCCCCE0078- <1>     db  03eh, 06ch, 0cch, 0feh, 0cch, 0cch, 0ceh, 000h, 078h, 0cch, 000h, 078h, 0cch, 0cch, 078h, 000h
3653 00015BD5 CC0078CCCC7800      <1>
3654 00015BDC 00CC0078CCCC780000- <1>     db  000h, 0cch, 000h, 078h, 0cch, 0cch, 078h, 000h, 000h, 0e0h, 000h, 078h, 0cch, 0cch, 078h, 000h
3654 00015BE5 E00078CCCC7800      <1>
3655 00015BEC 78CC00CCCCCC7E0000- <1>     db  078h, 0cch, 000h, 0cch, 0cch, 0cch, 07eh, 000h, 000h, 0e0h, 000h, 0cch, 0cch, 0cch, 07eh, 000h
3655 00015BF5 E000CCCCCC7E00      <1>
3656 00015BFC 00CC00CCCC7C0CF8C3- <1>     db  000h, 0cch, 000h, 0cch, 0cch, 07ch, 00ch, 0f8h, 0c3h, 018h, 03ch, 066h, 066h, 03ch, 018h, 000h
3656 00015C05 183C66663C1800      <1>
3657 00015C0C CC00CCCCCCCC780018- <1>     db  0cch, 000h, 0cch, 0cch, 0cch, 0cch, 078h, 000h, 018h, 018h, 07eh, 0c0h, 0c0h, 07eh, 018h, 018h
3657 00015C15 187EC0C07E1818      <1>
```

```
3658 00015C1C 386C64F060E6FC00CC- <1>     db  038h, 06ch, 064h, 0f0h, 060h, 0e6h, 0fch, 000h, 0cch, 0cch, 078h, 0fch, 030h, 0fch, 030h, 030h
3658 00015C25 CC78FC30FC3030      <1>
3659 00015C2C F8CCCCFAC6CFC6C70E- <1>     db  0f8h, 0cch, 0cch, 0fah, 0c6h, 0cfh, 0c6h, 0c7h, 00eh, 01bh, 018h, 03ch, 018h, 018h, 0d8h, 070h
3659 00015C35 1B183C1818D870      <1>
3660 00015C3C 1C00780C7CCC7E0038- <1>     db  01ch, 000h, 078h, 00ch, 07ch, 0cch, 07eh, 000h, 038h, 000h, 070h, 030h, 030h, 030h, 078h, 000h
3660 00015C45 00703030307800      <1>
3661 00015C4C 001C0078CCCC780000- <1>     db  000h, 01ch, 000h, 078h, 0cch, 0cch, 078h, 000h, 000h, 01ch, 000h, 0cch, 0cch, 0cch, 07eh, 000h
3661 00015C55 1C00CCCCCC7E00      <1>
3662 00015C5C 00F800F8CCCCCC00FC- <1>     db  000h, 0f8h, 000h, 0f8h, 0cch, 0cch, 0cch, 000h, 0fch, 000h, 0cch, 0ech, 0fch, 0dch, 0cch, 000h
3662 00015C65 00CCECFCDCCC00      <1>
3663 00015C6C 3C6C6C3E007E000038- <1>     db  03ch, 06ch, 06ch, 03eh, 000h, 07eh, 000h, 000h, 038h, 06ch, 06ch, 038h, 000h, 07ch, 000h, 000h
3663 00015C75 6C6C38007C0000      <1>
3664 00015C7C 30003060C0CC780000- <1>     db  030h, 000h, 030h, 060h, 0c0h, 0cch, 078h, 000h, 000h, 000h, 000h, 0fch, 0c0h, 0c0h, 000h, 000h
3664 00015C85 0000FCC0C00000      <1>
3665 00015C8C 000000FC0C0C0000C3- <1>     db  000h, 000h, 000h, 0fch, 00ch, 00ch, 000h, 000h, 0c3h, 0c6h, 0cch, 0deh, 033h, 066h, 0cch, 00fh
3665 00015C95 C6CCDE3366CC0F      <1>
3666 00015C9C C3C6CCDB376FCF0318- <1>     db  0c3h, 0c6h, 0cch, 0dbh, 037h, 06fh, 0cfh, 003h, 018h, 018h, 000h, 018h, 018h, 018h, 018h, 000h
3666 00015CA5 18001818181800      <1>
3667 00015CAC 003366CC6633000000- <1>     db  000h, 033h, 066h, 0cch, 066h, 033h, 000h, 000h, 000h, 0cch, 066h, 033h, 066h, 0cch, 000h, 000h
3667 00015CB5 CC663366CC0000      <1>
3668 00015CBC 2288228822882288 55- <1>     db  022h, 088h, 022h, 088h, 022h, 088h, 022h, 088h, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah
3668 00015CC5 AA55AA55AA55AA      <1>
3669 00015CCC DB77DBEEDB77DBEE18- <1>     db  0dbh, 077h, 0dbh, 0eeh, 0dbh, 077h, 0dbh, 0eeh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
3669 00015CD5 18181818181818      <1>
3670 00015CDC 18181818F818181818- <1>     db  018h, 018h, 018h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 018h, 018h, 018h
3670 00015CE5 18F818F8181818      <1>
3671 00015CEC 36363636F636363600- <1>     db  036h, 036h, 036h, 036h, 0f6h, 036h, 036h, 036h, 000h, 000h, 000h, 000h, 0feh, 036h, 036h, 036h
3671 00015CF5 000000FE363636      <1>
3672 00015CFC 0000F818F818181836- <1>     db  000h, 000h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 036h, 036h, 0f6h, 006h, 0f6h, 036h, 036h, 036h
3672 00015D05 36F606F6363636      <1>
3673 00015D0C 363636363636363600- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 000h, 000h, 0feh, 006h, 0f6h, 036h, 036h, 036h
3673 00015D15 00FE06F6363636      <1>
3674 00015D1C 3636F606FE00000036- <1>     db  036h, 036h, 0f6h, 006h, 0feh, 000h, 000h, 000h, 036h, 036h, 036h, 036h, 0feh, 000h, 000h, 000h
3674 00015D25 363636FE000000      <1>
3675 00015D2C 1818F818F800000000- <1>     db  018h, 018h, 0f8h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 018h, 018h
3675 00015D35 000000F8181818      <1>
3676 00015D3C 181818181F00000018- <1>     db  018h, 018h, 018h, 018h, 01fh, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 0ffh, 000h, 000h, 000h
3676 00015D45 181818FF000000      <1>
3677 00015D4C 00000000FF18181818- <1>     db  000h, 000h, 000h, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 018h, 018h
3677 00015D55 1818181F181818      <1>
3678 00015D5C 00000000FF00000018- <1>     db  000h, 000h, 000h, 000h, 0ffh, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h
3678 00015D65 181818FF181818      <1>
3679 00015D6C 18181F181F18181836- <1>     db  018h, 018h, 01fh, 018h, 01fh, 018h, 018h, 018h, 036h, 036h, 036h, 036h, 037h, 036h, 036h, 036h
3679 00015D75 36363637363636      <1>
3680 00015D7C 363637303F00000000- <1>     db  036h, 036h, 037h, 030h, 03fh, 000h, 000h, 000h, 000h, 000h, 03fh, 030h, 037h, 036h, 036h, 036h
3680 00015D85 003F3037363636      <1>
3681 00015D8C 3636F700FF00000000- <1>     db  036h, 036h, 0f7h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0f7h, 036h, 036h, 036h
3681 00015D95 00FF00F7363636      <1>
3682 00015D9C 363637303736363600- <1>     db  036h, 036h, 037h, 030h, 037h, 036h, 036h, 036h, 000h, 000h, 0ffh, 000h, 0ffh, 000h, 000h, 000h
3682 00015DA5 00FF00FF000000      <1>
3683 00015DAC 3636F700F736363618- <1>     db  036h, 036h, 0f7h, 000h, 0f7h, 036h, 036h, 036h, 018h, 018h, 0ffh, 000h, 0ffh, 000h, 000h, 000h
3683 00015DB5 18FF00FF000000      <1>
3684 00015DBC 36363636FF00000000- <1>     db  036h, 036h, 036h, 036h, 0ffh, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 018h, 018h, 018h
3684 00015DC5 00FF00FF181818      <1>
3685 00015DCC 00000000FF36363636- <1>     db  000h, 000h, 000h, 000h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 03fh, 000h, 000h, 000h
3685 00015DD5 3636363F000000      <1>
3686 00015DDC 18181F181F00000000- <1>     db  018h, 018h, 01fh, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 01fh, 018h, 01fh, 018h, 018h, 018h
3686 00015DE5 001F181F181818      <1>
3687 00015DEC 000000003F36363636- <1>     db  000h, 000h, 000h, 000h, 03fh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 036h, 036h, 036h
3687 00015DF5 363636FF363636      <1>
3688 00015DFC 1818FF18FF18181818- <1>     db  018h, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 000h, 000h, 000h
3688 00015E05 181818F8000000      <1>
3689 00015E0C 000000001F181818FF- <1>     db  000h, 000h, 000h, 000h, 01fh, 018h, 018h, 018h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
3689 00015E15 FFFFFFFFFFFFFF      <1>
3690 00015E1C 00000000FFFFFFFFF0- <1>     db  000h, 000h, 000h, 000h, 0ffh, 0ffh, 0ffh, 0ffh, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h
3690 00015E25 F0F0F0F0F0F0F0      <1>
3691 00015E2C 0F0F0F0F0F0F0F0FFF- <1>     db  00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h
3691 00015E35 FFFFFF00000000      <1>
3692 00015E3C 000076DCC8DC760000- <1>     db  000h, 000h, 076h, 0dch, 0c8h, 0dch, 076h, 000h, 000h, 078h, 0cch, 0f8h, 0cch, 0f8h, 0c0h, 0c0h
3692 00015E45 78CCF8CCF8C0C0      <1>
3693 00015E4C 00FCCCC0C0C0C00000- <1>     db  000h, 0fch, 0cch, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 0feh, 06ch, 06ch, 06ch, 06ch, 06ch, 000h
3693 00015E55 FE6C6C6C6C6C00      <1>
3694 00015E5C FCCC603060CCFC0000- <1>     db  0fch, 0cch, 060h, 030h, 060h, 0cch, 0fch, 000h, 000h, 000h, 07eh, 0d8h, 0d8h, 0d8h, 070h, 000h
3694 00015E65 007ED8D8D87000      <1>
3695 00015E6C 00666666667C60C000- <1>     db  000h, 066h, 066h, 066h, 066h, 07ch, 060h, 0c0h, 000h, 076h, 0dch, 018h, 018h, 018h, 018h, 000h
3695 00015E75 76DC1818181800      <1>
3696 00015E7C FC3078CCCC7830FC38- <1>     db  0fch, 030h, 078h, 0cch, 0cch, 078h, 030h, 0fch, 038h, 06ch, 0c6h, 0feh, 0c6h, 06ch, 038h, 000h
3696 00015E85 6CC6FEC66C3800      <1>
3697 00015E8C 386CC6C66C6CEE001C- <1>     db  038h, 06ch, 0c6h, 0c6h, 06ch, 06ch, 0eeh, 000h, 01ch, 030h, 018h, 07ch, 0cch, 0cch, 078h, 000h
3697 00015E95 30187CCCCC7800      <1>
3698 00015E9C 00007EDBDB7E000006- <1>     db  000h, 000h, 07eh, 0dbh, 0dbh, 07eh, 000h, 000h, 006h, 00ch, 07eh, 0dbh, 0dbh, 07eh, 060h, 0c0h
3698 00015EA5 0C7EDBDB7E60C0      <1>
3699 00015EAC 3860C0F8C060380078- <1>     db  038h, 060h, 0c0h, 0f8h, 0c0h, 060h, 038h, 000h, 078h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 000h
3699 00015EB5 CCCCCCCCCCCC00      <1>
3700 00015EBC 00FC00FC00FC000030- <1>     db  000h, 0fch, 000h, 0fch, 000h, 0fch, 000h, 000h, 030h, 030h, 0fch, 030h, 030h, 000h, 0fch, 000h
3700 00015EC5 30FC303000FC00      <1>
3701 00015ECC 603018306000FC0018- <1>     db  060h, 030h, 018h, 030h, 060h, 000h, 0fch, 000h, 018h, 030h, 060h, 030h, 018h, 000h, 0fch, 000h
3701 00015ED5 3060301800FC00      <1>
3702 00015EDC 0E1B1B181818181818- <1>     db  00eh, 01bh, 01bh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0d8h, 0d8h, 070h
3702 00015EE5 18181818D8D870      <1>
3703 00015EEC 303000FC0030300000- <1>     db  030h, 030h, 000h, 0fch, 000h, 030h, 030h, 000h, 000h, 076h, 0dch, 000h, 076h, 0dch, 000h, 000h
3703 00015EF5 76DC0076DC0000      <1>
3704 00015EFC 386C6C3800000000- <1>     db  038h, 06ch, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h
3704 00015F05 00001818000000      <1>
3705 00015F0C 000000001800000F- <1>     db  000h, 000h, 000h, 000h, 018h, 000h, 000h, 000h, 00fh, 00ch, 00ch, 00ch, 0ech, 06ch, 03ch, 01ch
3705 00015F15 0C0C0CEC6C3C1C      <1>
3706 00015F1C 786C6C6C6C00000070- <1>     db  078h, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 070h, 018h, 030h, 060h, 078h, 000h, 000h, 000h
3706 00015F25 18306078000000      <1>
3707 00015F2C 00003C3C3C3C000000- <1>     db  000h, 000h, 03ch, 03ch, 03ch, 03ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3707 00015F35 000000000000000     <1>
3708                              <1> vgafont14:
3709 00015F3C 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3709 00015F45 00000000000000      <1>
3710 00015F4C 7E81A58181BD99817E- <1>     db  07eh, 081h, 0a5h, 081h, 081h, 0bdh, 099h, 081h, 07eh, 000h, 000h, 000h, 000h, 000h, 07eh, 0ffh
3710 00015F55 000000000007EFF     <1>
3711 00015F5C DBFFFFC3E7FF7E0000- <1>     db  0dbh, 0ffh, 0ffh, 0c3h, 0e7h, 0ffh, 07eh, 000h, 000h, 000h, 000h, 000h, 000h, 06ch, 0feh, 0feh
3711 00015F65 000000006CFEFE      <1>
3712 00015F6C FEFE7C381000000000- <1>     db  0feh, 0feh, 07ch, 038h, 010h, 000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 07ch, 0feh, 07ch
3712 00015F75 000010387CFE7C      <1>
3713 00015F7C 3810000000000000018- <1>     db  038h, 010h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 03ch, 0e7h, 0e7h, 0e7h, 018h, 018h
3713 00015F85 3C3CE7E7E71818      <1>
```

```
3714 00015F8C 3C0000000000183C7E- <1>     db  03ch, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 07eh, 0ffh, 0ffh, 07eh, 018h, 018h, 03ch, 000h
3714 00015F95 FFFF7E18183C00      <1>
3715 00015F9C 00000000000000183C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 03ch, 018h, 000h, 000h, 000h, 000h, 000h
3715 00015FA5 3C180000000000      <1>
3716 00015FAC FFFFFFFFFFE7C3C3E7- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0e7h, 0c3h, 0c3h, 0e7h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h
3716 00015FB5 FFFFFFFFFF0000      <1>
3717 00015FBC 00003C664242663C00- <1>     db  000h, 000h, 03ch, 066h, 042h, 042h, 066h, 03ch, 000h, 000h, 000h, 000h, 0ffh, 0ffh, 0ffh, 0ffh
3717 00015FC5 000000FFFFFFFF      <1>
3718 00015FCC C399BDBD99C3FFFFFF- <1>     db  0c3h, 099h, 0bdh, 0bdh, 099h, 0c3h, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 01eh, 00eh, 01ah, 032h
3718 00015FD5 FF00001E0E1A32      <1>
3719 00015FDC 78CCCCCC7800000000- <1>     db  078h, 0cch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h, 000h, 03ch, 066h, 066h, 066h, 03ch, 018h
3719 00015FE5 003C6666663C18      <1>
3720 00015FEC 7E1818000000000003F- <1>     db  07eh, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 03fh, 033h, 03fh, 030h, 030h, 030h, 070h, 0f0h
3720 00015FF5 333F30303070F0      <1>
3721 00015FFC E0000000000007F637F- <1>     db  0e0h, 000h, 000h, 000h, 000h, 000h, 07fh, 063h, 07fh, 063h, 063h, 063h, 067h, 0e7h, 0e6h, 0c0h
3721 00016005 63636367E7E6C0      <1>
3722 0001600C 000000001818DB3CE7- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 0dbh, 03ch, 0e7h, 03ch, 0dbh, 018h, 018h, 000h, 000h, 000h
3722 00016015 3CDB1818000000      <1>
3723 0001601C 000080C0E0F8FEF8E0- <1>     db  000h, 000h, 080h, 0c0h, 0e0h, 0f8h, 0feh, 0f8h, 0e0h, 0c0h, 080h, 000h, 000h, 000h, 000h, 000h
3723 00016025 C0800000000000      <1>
3724 0001602C 02060E3EFE3E0E0602- <1>     db  002h, 006h, 00eh, 03eh, 0feh, 03eh, 00eh, 006h, 002h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch
3724 00016035 0000000000183C      <1>
3725 0001603C 7E1818187E3C180000- <1>     db  07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h
3725 00016045 00000066666666      <1>
3726 0001604C 666600666600000000- <1>     db  066h, 066h, 000h, 066h, 066h, 000h, 000h, 000h, 000h, 000h, 07fh, 0dbh, 0dbh, 0dbh, 07bh, 01bh
3726 00016055 007FDBDBDB7B1B      <1>
3727 0001605C 1B1B1B000000007CC6- <1>     db  01bh, 01bh, 01bh, 000h, 000h, 000h, 000h, 07ch, 0c6h, 060h, 038h, 06ch, 0c6h, 0c6h, 06ch, 038h
3727 00016065 60386CC6C66C38      <1>
3728 0001606C 0CC67C0000000000000- <1>     db  00ch, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 0feh, 0feh, 000h
3728 00016075 000000FEFEFE00      <1>
3729 0001607C 00000000183C7E1818- <1>     db  000h, 000h, 000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 07eh, 000h, 000h
3729 00016085 187E3C187E0000      <1>
3730 0001608C 0000183C7E18181818- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h
3730 00016095 18180000000000      <1>
3731 0001609C 1818181818187E3C18- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3731 000160A5 00000000000000      <1>
3732 000160AC 180CFE0C1800000000- <1>     db  018h, 00ch, 0feh, 00ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 030h, 060h
3732 000160B5 00000000003060      <1>
3733 000160BC FE6030000000000000- <1>     db  0feh, 060h, 030h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0c0h, 0c0h, 0c0h
3733 000160C5 00000000C0C0C0      <1>
3734 000160CC FE00000000000000000- <1>     db  0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 028h, 06ch, 0feh, 06ch, 028h, 000h
3734 000160D5 00286CFE6C2800      <1>
3735 000160DC 000000000000001038- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 038h, 07ch, 07ch, 0feh, 0feh, 000h, 000h
3735 000160E5 387C7CFEFE0000      <1>
3736 000160EC 0000000000FEFE7C7C- <1>     db  000h, 000h, 000h, 000h, 000h, 0feh, 0feh, 07ch, 07ch, 038h, 038h, 010h, 000h, 000h, 000h, 000h
3736 000160F5 38381000000000      <1>
3737 000160FC 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3737 00016105 00000000000000      <1>
3738 0001610C 183C3C3C1818001818- <1>     db  018h, 03ch, 03ch, 03ch, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 066h, 066h, 066h
3738 00016115 00000000666666      <1>
3739 0001611C 240000000000000000- <1>     db  024h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 06ch, 06ch, 0feh, 06ch
3739 00016125 0000006C6CFE6C      <1>
3740 0001612C 6C6CFE6C6C00000018- <1>     db  06ch, 06ch, 0feh, 06ch, 06ch, 000h, 000h, 000h, 018h, 018h, 07ch, 0c6h, 0c2h, 0c0h, 07ch, 006h
3740 00016135 187CC6C2C07C06      <1>
3741 0001613C 86C67C181800000000- <1>     db  086h, 0c6h, 07ch, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 0c2h, 0c6h, 00ch, 018h, 030h, 066h
3741 00016145 00C2C60C183066      <1>
3742 0001614C C600000000000386C6C- <1>     db  0c6h, 000h, 000h, 000h, 000h, 000h, 038h, 06ch, 06ch, 038h, 076h, 0dch, 0cch, 0cch, 076h, 000h
3742 00016155 3876DCCCCC7600      <1>
3743 0001615C 000003030306000000- <1>     db  000h, 000h, 000h, 030h, 030h, 030h, 060h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3743 00016165 00000000000000      <1>
3744 0001616C 00000C183030303030- <1>     db  000h, 000h, 00ch, 018h, 030h, 030h, 030h, 030h, 030h, 018h, 00ch, 000h, 000h, 000h, 000h, 000h
3744 00016175 180C0000000000      <1>
3745 0001617C 30180C0C0C0C0C1830- <1>     db  030h, 018h, 00ch, 00ch, 00ch, 00ch, 00ch, 018h, 030h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3745 00016185 00000000000000      <1>
3746 0001618C 663CFF3C66000000000- <1>     db  066h, 03ch, 0ffh, 03ch, 066h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h
3746 00016195 00000000001818      <1>
3747 0001619C 7E1818000000000000- <1>     db  07eh, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3747 000161A5 00000000000000      <1>
3748 000161AC 181818300000000000- <1>     db  018h, 018h, 018h, 030h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 000h, 000h, 000h
3748 000161B5 000000FE000000      <1>
3749 000161BC 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h
3749 000161C5 00000000181800      <1>
3750 000161CC 0000000002060C1830- <1>     db  000h, 000h, 000h, 000h, 002h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 080h, 000h, 000h, 000h, 000h
3750 000161D5 60C08000000000      <1>
3751 000161DC 00007CC6CEDEF6E6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0ceh, 0deh, 0f6h, 0e6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h
3751 000161E5 C67C0000000000      <1>
3752 000161EC 183878181818187E- <1>     db  018h, 038h, 078h, 018h, 018h, 018h, 018h, 018h, 07eh, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h
3752 000161F5 00000000007CC6      <1>
3753 000161FC 060C183060C6FE0000- <1>     db  006h, 00ch, 018h, 030h, 060h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 006h, 006h
3753 00016205 0000007CC60606      <1>
3754 0001620C 3C0606C67C00000000- <1>     db  03ch, 006h, 006h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 00ch, 01ch, 03ch, 06ch, 0cch, 0feh
3754 00016215 000C1C3C6CCCFE      <1>
3755 0001621C 0C0C1E00000000000FE- <1>     db  00ch, 00ch, 01eh, 000h, 000h, 000h, 000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 0fch, 006h, 006h, 0c6h
3755 00016225 C0C0C0FC0606C6      <1>
3756 0001622C 7C00000000003860C0- <1>     db  07ch, 000h, 000h, 000h, 000h, 000h, 038h, 060h, 0c0h, 0c0h, 0fch, 0c6h, 0c6h, 0c6h, 07ch, 000h
3756 00016235 C0FCC6C6C67C00      <1>
3757 0001623C 00000000FEC6060C18- <1>     db  000h, 000h, 000h, 000h, 0feh, 0c6h, 006h, 00ch, 018h, 030h, 030h, 030h, 030h, 000h, 000h, 000h
3757 00016245 30303030000000      <1>
3758 0001624C 00007CC6C6C67CC6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 07ch, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h
3758 00016255 C67C0000000000      <1>
3759 0001625C 7CC6C6C67E06060C78- <1>     db  07ch, 0c6h, 0c6h, 0c6h, 07eh, 006h, 006h, 00ch, 078h, 000h, 000h, 000h, 000h, 000h, 000h, 018h
3759 00016265 00000000000018      <1>
3760 0001626C 1800000018180000000- <1>     db  018h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h
3760 00016275 00000000181800      <1>
3761 0001627C 000018183000000000- <1>     db  000h, 000h, 018h, 018h, 030h, 000h, 000h, 000h, 000h, 000h, 006h, 00ch, 018h, 030h, 060h, 030h
3761 00016285 00060C18306030      <1>
3762 0001628C 180C060000000000000- <1>     db  018h, 00ch, 006h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 000h, 000h, 07eh, 000h
3762 00016295 00007E00007E00      <1>
3763 0001629C 000000000000603018- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 00ch, 006h, 00ch, 018h, 030h, 060h, 000h
3763 000162A5 0C060C18306000      <1>
3764 000162AC 000000007CC6C60C18- <1>     db  000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 00ch, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h
3764 000162B5 18001818000000      <1>
3765 000162BC 00007CC6C6DEDEDEDC- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0deh, 0deh, 0deh, 0dch, 0c0h, 07ch, 000h, 000h, 000h, 000h, 000h
3765 000162C5 C07C0000000000      <1>
3766 000162CC 10386CC6C6FEC6C6C6- <1>     db  010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h, 000h, 0fch, 066h
3766 000162D5 0000000000FC66      <1>
3767 000162DC 66667C666666FC0000- <1>     db  066h, 066h, 07ch, 066h, 066h, 066h, 0fch, 000h, 000h, 000h, 000h, 000h, 03ch, 066h, 0c2h, 0c0h
3767 000162E5 0000003C66C2C0      <1>
3768 000162EC C0C0C2663C00000000- <1>     db  0c0h, 0c0h, 0c2h, 066h, 03ch, 000h, 000h, 000h, 000h, 000h, 0f8h, 06ch, 066h, 066h, 066h, 066h
3768 000162F5 00F86C66666666      <1>
3769 000162FC 666CF800000000000FE- <1>     db  066h, 06ch, 0f8h, 000h, 000h, 000h, 000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 062h, 066h
```

```
3769 00016305 66626878686266      <1>
3770 0001630C FE0000000000FE6662- <1>     db  0feh, 000h, 000h, 000h, 000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 060h, 060h, 0f0h, 000h
3770 00016315 6878686060F000      <1>
3771 0001631C 000000003C66C2C0C0- <1>     db  000h, 000h, 000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0deh, 0c6h, 066h, 03ah, 000h, 000h, 000h
3771 00016325 DEC6663A000000      <1>
3772 0001632C 0000C6C6C6C6FEC6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h, 000h
3772 00016335 C6C600000000000     <1>
3773 0001633C 3C181818181818183C- <1>     db  03ch, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 01eh, 00ch
3773 00016345 00000000001E0C      <1>
3774 0001634C 0C0C0C0CCCCC780000- <1>     db  00ch, 00ch, 00ch, 00ch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h, 000h, 0e6h, 066h, 06ch, 06ch
3774 00016355 000000E6666C6C      <1>
3775 0001635C 786C6C66E600000000- <1>     db  078h, 06ch, 06ch, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h, 0f0h, 060h, 060h, 060h, 060h, 060h
3775 00016365 00F06060606060      <1>
3776 0001636C 6266FE0000000000C6- <1>     db  062h, 066h, 0feh, 000h, 000h, 000h, 000h, 000h, 0c6h, 0eeh, 0feh, 0feh, 0d6h, 0c6h, 0c6h, 0c6h
3776 00016375 EEFEFED6C6C6C6      <1>
3777 0001637C C600000000000C6E6F6- <1>     db  0c6h, 000h, 000h, 000h, 000h, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h, 0c6h, 0c6h, 000h
3777 00016385 FEDECEC6C6C600      <1>
3778 0001638C 00000000386CC6C6C6- <1>     db  000h, 000h, 000h, 000h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h
3778 00016395 C6C66C38000000      <1>
3779 0001639C 0000FC6666667C6060- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h, 000h
3779 000163A5 60F000000000000     <1>
3780 000163AC 7CC6C6C6C6D6DE7C0C- <1>     db  07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0d6h, 0deh, 07ch, 00ch, 00eh, 000h, 000h, 000h, 000h, 0fch, 066h
3780 000163B5 0E00000000FC66      <1>
3781 000163BC 66667C6C6666E60000- <1>     db  066h, 066h, 07ch, 06ch, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 060h
3781 000163C5 0000007CC6C660      <1>
3782 000163CC 380CC6C67C00000000- <1>     db  038h, 00ch, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 07eh, 07eh, 05ah, 018h, 018h, 018h
3782 000163D5 007E7E5A181818      <1>
3783 000163DC 18183C00000000C6- <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h
3783 000163E5 C6C6C6C6C6C6C6      <1>
3784 000163EC 7C0000000000C6C6C6- <1>     db  07ch, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 06ch, 038h, 010h, 000h
3784 000163F5 C6C6C66C381000      <1>
3785 000163FC 00000000C6C6C6C6D6- <1>     db  000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0d6h, 0d6h, 0feh, 07ch, 06ch, 000h, 000h, 000h
3785 00016405 D6FE7C6C000000      <1>
3786 0001640C 0000C6C66C3838386C- <1>     db  000h, 000h, 0c6h, 0c6h, 06ch, 038h, 038h, 038h, 06ch, 0c6h, 0c6h, 000h, 000h, 000h, 000h, 000h
3786 00016415 C6C600000000000     <1>
3787 0001641C 666666663C1818183C- <1>     db  066h, 066h, 066h, 066h, 03ch, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 0feh, 0c6h
3787 00016425 0000000000FEC6      <1>
3788 0001642C 8C183060C2C6FE0000- <1>     db  08ch, 018h, 030h, 060h, 0c2h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h, 03ch, 030h, 030h, 030h
3788 00016435 0000003C303030      <1>
3789 0001643C 303030303C00000000- <1>     db  030h, 030h, 030h, 030h, 03ch, 000h, 000h, 000h, 000h, 000h, 080h, 0c0h, 0e0h, 070h, 038h, 01ch
3789 00016445 0080C0E070381C      <1>
3790 0001644C 0E060200000000003C- <1>     db  00eh, 006h, 002h, 000h, 000h, 000h, 000h, 000h, 03ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch
3790 00016455 0C0C0C0C0C0C0C      <1>
3791 0001645C 3C00000010386CC600- <1>     db  03ch, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3791 00016465 000000000000000     <1>
3792 0001646C 0000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h
3792 00016475 0000000000FF00      <1>
3793 0001647C 3030180000000000000- <1>     db  030h, 030h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3793 00016485 000000000000000     <1>
3794 0001648C 000000780C7CCCCC76- <1>     db  000h, 000h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 0e0h, 060h
3794 00016495 0000000000E060      <1>
3795 0001649C 60786C6666667C0000- <1>     db  060h, 078h, 06ch, 066h, 066h, 066h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch
3795 000164A5 0000000000007C      <1>
3796 000164AC C6C0C0C67C00000000- <1>     db  0c6h, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 01ch, 00ch, 00ch, 03ch, 06ch, 0cch
3796 000164B5 001C0C0C3C6CCC      <1>
3797 000164BC CCCC760000000000000- <1>     db  0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h
3797 000164C5 00007CC6FEC0C6      <1>
3798 000164CC 7C0000000000386C64- <1>     db  07ch, 000h, 000h, 000h, 000h, 000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h, 060h, 0f0h, 000h
3798 000164D5 60F0606060F000      <1>
3799 000164DC 0000000000000076CC- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 07ch, 00ch, 0cch, 078h, 000h
3799 000164E5 CCCC7C0CCC7800      <1>
3800 000164EC 0000E060606C766666- <1>     db  000h, 000h, 0e0h, 060h, 060h, 06ch, 076h, 066h, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h
3800 000164F5 66E600000000000     <1>
3801 000164FC 18180038181818183C- <1>     db  018h, 018h, 000h, 038h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 006h, 006h
3801 00016505 000000000000606     <1>
3802 0001650C 000E0606060666663C- <1>     db  000h, 00eh, 006h, 006h, 006h, 006h, 066h, 066h, 03ch, 000h, 000h, 000h, 0e0h, 060h, 060h, 066h
3802 00016515 000000E0606066      <1>
3803 0001651C 6C786C66E600000000- <1>     db  06ch, 078h, 06ch, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h, 038h, 018h, 018h, 018h, 018h, 018h
3803 00016525 00381818181818      <1>
3804 0001652C 18183C000000000000- <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ech, 0feh, 0d6h, 0d6h, 0d6h
3804 00016535 0000ECFED6D6D6      <1>
3805 0001653C C600000000000000000- <1>     db  0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 000h
3805 00016545 DC666666666600      <1>
3806 0001654C 000000000000007CC6- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h
3806 00016555 C6C6C67C000000      <1>
3807 0001655C 0000000000DC666666- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 07ch, 060h, 060h, 0f0h, 000h, 000h, 000h
3807 00016565 7C6060F0000000      <1>
3808 0001656C 00000076CCCCCC7C0C- <1>     db  000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 07ch, 00ch, 00ch, 01eh, 000h, 000h, 000h, 000h, 000h
3808 00016575 0C1E0000000000      <1>
3809 0001657C 00DC76666060F00000- <1>     db  000h, 0dch, 076h, 066h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch
3809 00016585 0000000000007C      <1>
3810 0001658C C6701CC67C00000000- <1>     db  0c6h, 070h, 01ch, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 010h, 030h, 030h, 0fch, 030h, 030h
3810 00016595 00103030FC3030      <1>
3811 0001659C 30361C0000000000000- <1>     db  030h, 036h, 01ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch
3811 000165A5 0000CCCCCCCCCC      <1>
3812 000165AC 760000000000000000- <1>     db  076h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h, 03ch, 018h, 000h
3812 000165B5 666666663C1800      <1>
3813 000165BC 00000000000000C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0d6h, 0d6h, 0feh, 06ch, 000h, 000h, 000h
3813 000165C5 D6D6FE6C000000      <1>
3814 000165CC 0000000000C66C3838- <1>     db  000h, 000h, 000h, 000h, 000h, 0c6h, 06ch, 038h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h
3814 000165D5 6CC600000000000     <1>
3815 000165DC 000000C6C6C6C67E06- <1>     db  000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 0f8h, 000h, 000h, 000h, 000h, 000h
3815 000165E5 0CF800000000000     <1>
3816 000165EC 00FECC183066FE0000- <1>     db  000h, 0feh, 0cch, 018h, 030h, 066h, 0feh, 000h, 000h, 000h, 000h, 000h, 00eh, 018h, 018h, 018h
3816 000165F5 0000000E181818      <1>
3817 000165FC 701818180E00000000- <1>     db  070h, 018h, 018h, 018h, 00eh, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 000h, 018h
3817 00016605 00181818180018      <1>
3818 0001660C 1818180000000000070- <1>     db  018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 070h, 018h, 018h, 018h, 00eh, 018h, 018h, 018h
3818 00016615 1818180E181818      <1>
3819 0001661C 700000000000076DC00- <1>     db  070h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3819 00016625 000000000000000     <1>
3820 0001662C 0000000000010386C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 000h, 000h, 000h, 000h
3820 00016635 C6C6FE00000000      <1>
3821 0001663C 00003C66C2C0C0C266- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0c2h, 066h, 03ch, 00ch, 006h, 07ch, 000h, 000h, 000h
3821 00016645 3C0C067C000000      <1>
3822 0001664C CCCC00CCCCCCCCCC76- <1>     db  0cch, 0cch, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 00ch, 018h, 030h
3822 00016655 000000000C1830      <1>
3823 0001665C 007CC6FEC0C67C0000- <1>     db  000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 000h, 078h
3823 00016665 000010386C0078      <1>
3824 0001666C 0C7CCCCC7600000000- <1>     db  00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 0cch, 0cch, 000h, 078h, 00ch, 07ch
3824 00016675 00CCCC00780C7C      <1>
```

```
3825 0001667C CCCC76000000006030- <1>     db  0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 000h, 078h, 00ch, 07ch, 0cch, 0cch
3825 00016685 1800780C7CCCCC      <1>
3826 0001668C 7600000000386C3800- <1>     db  076h, 000h, 000h, 000h, 000h, 038h, 06ch, 038h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 076h, 000h
3826 00016695 780C7CCCCC7600      <1>
3827 0001669C 0000000000003C6660- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 03ch, 066h, 060h, 066h, 03ch, 00ch, 006h, 03ch, 000h, 000h
3827 000166A5 663C0C063C0000      <1>
3828 000166AC 0010386C007CC6FEC0- <1>     db  000h, 010h, 038h, 06ch, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h
3828 000166B5 C67C0000000000      <1>
3829 000166BC CCCC007CC6FEC0C67C- <1>     db  0cch, 0cch, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 060h, 030h, 018h
3829 000166C5 00000000603018      <1>
3830 000166CC 007CC6FEC0C67C0000- <1>     db  000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 000h, 038h
3830 000166D5 00000066660038      <1>
3831 000166DC 181818183C00000000- <1>     db  018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 018h, 03ch, 066h, 000h, 038h, 018h, 018h
3831 000166E5 183C6600381818      <1>
3832 000166EC 18183C000000006030- <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 000h, 038h, 018h, 018h, 018h, 018h
3832 000166F5 18003818181818      <1>
3833 000166FC 3C00000000C6C61038- <1>     db  03ch, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 000h
3833 00016705 6CC6C6FEC6C600      <1>
3834 0001670C 0000386C3800386CC6- <1>     db  000h, 000h, 038h, 06ch, 038h, 000h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 000h, 000h, 000h
3834 00016715 C6FEC6C6000000      <1>
3835 0001671C 18306000FE66607C60- <1>     db  018h, 030h, 060h, 000h, 0feh, 066h, 060h, 07ch, 060h, 066h, 0feh, 000h, 000h, 000h, 000h, 000h
3835 00016725 66FE0000000000      <1>
3836 0001672C 0000CC76367ED8D86E- <1>     db  000h, 000h, 0cch, 076h, 036h, 07eh, 0d8h, 0d8h, 06eh, 000h, 000h, 000h, 000h, 000h, 03eh, 06ch
3836 00016735 00000000003E6C      <1>
3837 0001673C CCCCFECCCCCCCE0000- <1>     db  0cch, 0cch, 0feh, 0cch, 0cch, 0cch, 0ceh, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 000h, 07ch
3837 00016745 000010386C007C      <1>
3838 0001674C C6C6C6C67C00000000- <1>     db  0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 000h, 07ch, 0c6h, 0c6h
3838 00016755 00C6C6007CC6C6      <1>
3839 0001675C C6C67C000000006030- <1>     db  0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h
3839 00016765 18007CC6C6C6C6      <1>
3840 0001676C 7C000000003078CC00- <1>     db  07ch, 000h, 000h, 000h, 000h, 030h, 078h, 0cch, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h
3840 00016775 CCCCCCCCCC7600      <1>
3841 0001677C 00000060301800CCCC- <1>     db  000h, 000h, 000h, 060h, 030h, 018h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h
3841 00016785 CCCCCC76000000      <1>
3842 0001678C 0000C6C600C6C6C6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 078h, 000h, 000h, 0c6h
3842 00016795 7E060C780000C6      <1>
3843 0001679C C6386CC6C6C6C66C38- <1>     db  0c6h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 000h
3843 000167A5 00000000C6C600      <1>
3844 000167AC C6C6C6C6C6C67C0000- <1>     db  0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 018h, 018h, 03ch, 066h, 060h
3844 000167B5 000018183C6660      <1>
3845 000167BC 60663C1818000000000- <1>    db  060h, 066h, 03ch, 018h, 018h, 000h, 000h, 000h, 000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h
3845 000167C5 386C6460F06060      <1>
3846 000167CC 60E6FC000000000066- <1>     db  060h, 0e6h, 0fch, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 03ch, 018h, 07eh, 018h, 07eh, 018h
3846 000167D5 663C187E187E18      <1>
3847 000167DC 1800000000F8CCCCF8- <1>     db  018h, 000h, 000h, 000h, 000h, 0f8h, 0cch, 0cch, 0f8h, 0c4h, 0cch, 0deh, 0cch, 0cch, 0c6h, 000h
3847 000167E5 C4CCDECCCCC600      <1>
3848 000167EC 0000000E1B1818187E- <1>     db  000h, 000h, 000h, 00eh, 01bh, 018h, 018h, 018h, 07eh, 018h, 018h, 018h, 018h, 0d8h, 070h, 000h
3848 000167F5 18181818D87000      <1>
3849 000167FC 0018306000780C7CCC- <1>     db  000h, 018h, 030h, 060h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 00ch
3849 00016805 CC760000000000C     <1>
3850 0001680C 183000381818183C- <1>     db  018h, 030h, 000h, 038h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 018h, 030h, 060h
3850 00016815 00000000183060      <1>
3851 0001681C 007CC6C6C6C67C0000- <1>     db  000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 018h, 030h, 060h, 000h, 0cch
3851 00016825 000018306000CC      <1>
3852 0001682C CCCCCCCC7600000000- <1>     db  0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 0dch, 066h, 066h
3852 00016835 0076DC00DC6666      <1>
3853 0001683C 66666600000076DC00- <1>     db  066h, 066h, 066h, 000h, 000h, 000h, 076h, 0dch, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h
3853 00016845 C6E6F6FEDECEC6      <1>
3854 0001684C C6000000003C6C6C3E- <1>     db  0c6h, 000h, 000h, 000h, 000h, 03ch, 06ch, 06ch, 03eh, 000h, 07eh, 000h, 000h, 000h, 000h, 000h
3854 00016855 007E0000000000      <1>
3855 0001685C 000000386C6C38007C- <1>     db  000h, 000h, 000h, 038h, 06ch, 06ch, 038h, 000h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3855 00016865 00000000000000      <1>
3856 0001686C 0000303000303060C6- <1>     db  000h, 000h, 030h, 030h, 000h, 030h, 030h, 060h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h
3856 00016875 C67C0000000000      <1>
3857 0001687C 00000000FEC0C0C000- <1>     db  000h, 000h, 000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3857 00016885 00000000000000      <1>
3858 0001688C 0000FE0606060000000- <1>    db  000h, 000h, 0feh, 006h, 006h, 006h, 000h, 000h, 000h, 000h, 000h, 0c0h, 0c0h, 0c6h, 0cch, 0d8h
3858 00016895 0000C0C0C6CCD8      <1>
3859 0001689C 3060DC860C183E0000- <1>     db  030h, 060h, 0dch, 086h, 00ch, 018h, 03eh, 000h, 000h, 0c0h, 0c0h, 0c6h, 0cch, 0d8h, 030h, 066h
3859 000168A5 C0C0C6CCD83066      <1>
3860 000168AC CE9E3E0606000000018- <1>    db  0ceh, 09eh, 03eh, 006h, 006h, 000h, 000h, 000h, 018h, 018h, 000h, 018h, 018h, 03ch, 03ch, 03ch
3860 000168B5 180018183C3C3C      <1>
3861 000168BC 180000000000000036- <1>     db  018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 036h, 06ch, 0d8h, 06ch, 036h, 000h, 000h, 000h
3861 000168C5 6CD86C36000000      <1>
3862 000168CC 000000000000D86C36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0d8h, 06ch, 036h, 06ch, 0d8h, 000h, 000h, 000h, 000h, 000h
3862 000168D5 6CD800000000000     <1>
3863 000168DC 1144114411441144411- <1>    db  011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 055h, 0aah
3863 000168E5 441144114455AA      <1>
3864 000168EC 55AA55AA55AA55AA55- <1>     db  055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 0ddh, 077h, 0ddh, 077h
3864 000168F5 AA55AADD77DD77      <1>
3865 000168FC DD77DD77DD77DD77DD- <1>     db  0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 018h, 018h, 018h, 018h, 018h, 018h
3865 00016905 77181818181818      <1>
3866 0001690C 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h
3866 00016915 181818181818F8      <1>
3867 0001691C 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 018h, 018h
3867 00016925 1818F818F81818      <1>
3868 0001692C 1818181836363636- <1>     db  018h, 018h, 018h, 018h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0f6h, 036h, 036h, 036h, 036h
3868 00016935 3636F636363636      <1>
3869 0001693C 3636000000000000- <1>     db  036h, 036h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 036h, 036h, 036h, 036h, 036h, 036h
3869 00016945 FE363636363636      <1>
3870 0001694C 0000000000F818F818- <1>     db  000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 036h, 036h
3870 00016955 18181818183636      <1>
3871 0001695C 363636F606F6363636- <1>     db  036h, 036h, 036h, 0f6h, 006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
3871 00016965 36363636363636      <1>
3872 0001696C 3636363636363636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 000h, 000h, 000h, 000h, 000h, 0feh
3872 00016975 3600000000000FE     <1>
3873 0001697C 06F6363636363636- <1>     db  006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0f6h, 006h, 0feh
3873 00016985 36363636F606FE      <1>
3874 0001698C 0000000000000363636- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0feh, 000h, 000h
3874 00016995 36363636FE0000      <1>
3875 0001699C 0000000018181818- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 000h, 000h, 000h, 000h
3875 000169A5 F818F800000000      <1>
3876 000169AC 0000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h
3876 000169B5 F8181818181818      <1>
3877 000169BC 181818181818181F00- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h
3877 000169C5 00000000001818      <1>
3878 000169CC 1818181818FF000000- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3878 000169D5 00000000000000      <1>
3879 000169DC 000000FF1818181818- <1>     db  000h, 000h, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
3879 000169E5 18181818181818      <1>
3880 000169EC 181F181818181800- <1>     db  018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh
```

```
3880 000169F5 000000000000FF      <1>
3881 000169FC 00000000000181818- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 018h
3881 00016A05 18181818FF1818      <1>
3882 00016A0C 181818181818181818- <1>    db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh, 018h, 018h, 018h, 018h
3882 00016A15 1F181F18181818      <1>
3883 00016A1C 181836363636363636- <1>    db  018h, 018h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 037h, 036h, 036h, 036h, 036h, 036h, 036h
3883 00016A25 37363636363636      <1>
3884 00016A2C 363636363637303F00- <1>    db  036h, 036h, 036h, 036h, 036h, 037h, 030h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3884 00016A35 000000000000000     <1>
3885 00016A3C 0000003F3037363636- <1>    db  000h, 000h, 000h, 03fh, 030h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
3885 00016A45 36363636363636      <1>
3886 00016A4C 36F700FF0000000000- <1>    db  036h, 0f7h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh
3886 00016A55 000000000000FF      <1>
3887 00016A5C 00F7363636363636- <1>      db  000h, 0f7h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 037h, 030h, 037h
3887 00016A65 36363636373037      <1>
3888 00016A6C 363636363636000000- <1>    db  036h, 036h, 036h, 036h, 036h, 036h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 000h, 000h
3888 00016A75 0000FF00FF0000      <1>
3889 00016A7C 000000036363636363- <1>    db  000h, 000h, 000h, 000h, 036h, 036h, 036h, 036h, 036h, 0f7h, 000h, 0f7h, 036h, 036h, 036h, 036h
3889 00016A85 F700F736363636      <1>
3890 00016A8C 363618181818FF00- <1>      db  036h, 036h, 018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h
3890 00016A95 FF000000000000      <1>
3891 00016A9C 36363636363636FF00- <1>    db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3891 00016AA5 000000000000000     <1>
3892 00016AAC 000000FF00FF181818- <1>    db  000h, 000h, 000h, 0ffh, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h
3892 00016AB5 181818000000000     <1>
3893 00016ABC 000000FF3636363636- <1>    db  000h, 000h, 000h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
3893 00016AC5 36363636363636      <1>
3894 00016ACC 363F000000000000018- <1>   db  036h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh
3894 00016AD5 181818181F181F      <1>
3895 00016ADC 000000000000000000- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 01fh, 018h, 01fh, 018h, 018h
3895 00016AE5 00001F181F1818      <1>
3896 00016AEC 181818180000000000- <1>    db  018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 03fh, 036h, 036h, 036h, 036h
3896 00016AF5 00003F36363636      <1>
3897 00016AFC 363636363636363636- <1>    db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h
3897 00016B05 FF363636363636      <1>
3898 00016B0C 1818181818FF18FF18- <1>    db  018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
3898 00016B15 18181818181818      <1>
3899 00016B1C 1818181818F8000000- <1>    db  018h, 018h, 018h, 018h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3899 00016B25 000000000000000     <1>
3900 00016B2C 0000001F1818181818- <1>    db  000h, 000h, 000h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
3900 00016B35 18FFFFFFFFFFFF      <1>
3901 00016B3C FFFFFFFFFFFFFFFF00- <1>    db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh
3901 00016B45 000000000000FF      <1>
3902 00016B4C FFFFFFFFFFFFF0F0F0- <1>    db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h
3902 00016B55 F0F0F0F0F0F0F0      <1>
3903 00016B5C F0F0F0F00F0F0F0F0F- <1>    db  0f0h, 0f0h, 0f0h, 0f0h, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh
3903 00016B65 0F0F0F0F0F0F0F      <1>
3904 00016B6C 0F0FFFFFFFFFFFFFFF- <1>    db  00fh, 00fh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3904 00016B75 000000000000000     <1>
3905 00016B7C 000000000076DCD8D8- <1>    db  000h, 000h, 000h, 000h, 000h, 076h, 0dch, 0d8h, 0d8h, 0dch, 076h, 000h, 000h, 000h, 000h, 000h
3905 00016B85 DC760000000000      <1>
3906 00016B8C 00007CC6FCC6C6FCC0- <1>    db  000h, 000h, 07ch, 0c6h, 0fch, 0c6h, 0c6h, 0fch, 0c0h, 0c0h, 040h, 000h, 000h, 000h, 0feh, 0c6h
3906 00016B95 C040000000FEC6      <1>
3907 00016B9C C6C0C0C0C0C0C00000- <1>    db  0c6h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 06ch
3907 00016BA5 0000000000FE6C      <1>
3908 00016BAC 6C6C6C6C6C00000000- <1>    db  06ch, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 000h, 000h, 0feh, 0c6h, 060h, 030h, 018h, 030h
3908 00016BB5 00FEC660301830      <1>
3909 00016BBC 60C6FE000000000000- <1>    db  060h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 0d8h, 0d8h, 0d8h, 0d8h
3909 00016BC5 00007ED8D8D8D8      <1>
3910 00016BCC 700000000000000066- <1>    db  070h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h, 07ch, 060h, 060h, 0c0h
3910 00016BD5 6666667C6060C0      <1>
3911 00016BDC 00000000000076DC18- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h
3911 00016BE5 181818180000000     <1>
3912 00016BEC 00007E183C6666663C- <1>    db  000h, 000h, 07eh, 018h, 03ch, 066h, 066h, 066h, 03ch, 018h, 07eh, 000h, 000h, 000h, 000h, 000h
3912 00016BF5 187E0000000000      <1>
3913 00016BFC 386CC6C6FEC6C66C38- <1>    db  038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 038h, 06ch
3913 00016C05 0000000000386C      <1>
3914 00016C0C C6C6C66C6C6CEE0000- <1>    db  0c6h, 0c6h, 0c6h, 06ch, 06ch, 06ch, 0eeh, 000h, 000h, 000h, 000h, 000h, 01eh, 030h, 018h, 00ch
3914 00016C15 0000001E30180C      <1>
3915 00016C1C 3E6666663C00000000- <1>    db  03eh, 066h, 066h, 066h, 03ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 0dbh, 0dbh
3915 00016C25 000000007EDBDB      <1>
3916 00016C2C 7E00000000000000003- <1>   db  07eh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 003h, 006h, 07eh, 0dbh, 0dbh, 0f3h, 07eh, 060h
3916 00016C35 067EDBDBF37E60      <1>
3917 00016C3C C000000000001C3060- <1>    db  0c0h, 000h, 000h, 000h, 000h, 000h, 01ch, 030h, 060h, 060h, 07ch, 060h, 060h, 030h, 01ch, 000h
3917 00016C45 607C6060301C00      <1>
3918 00016C4C 00000000007CC6C6C6- <1>    db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h
3918 00016C55 C6C6C6C6000000      <1>
3919 00016C5C 000000FE0000FE0000- <1>    db  000h, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h
3919 00016C65 FE000000000000      <1>
3920 00016C6C 0018187E18180000FF- <1>    db  000h, 018h, 018h, 07eh, 018h, 018h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 030h, 018h
3920 00016C75 000000000003018     <1>
3921 00016C7C 0C060C1830007E0000- <1>    db  00ch, 006h, 00ch, 018h, 030h, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 00ch, 018h, 030h, 060h
3921 00016C85 0000000C183060      <1>
3922 00016C8C 30180C007E00000000- <1>    db  030h, 018h, 00ch, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 00eh, 01bh, 01bh, 018h, 018h, 018h
3922 00016C95 000E1B1B181818      <1>
3923 00016C9C 181818181818181818- <1>    db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0d8h, 0d8h
3923 00016CA5 1818181818D8D8      <1>
3924 00016CAC 700000000000001818- <1>    db  070h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 07eh, 000h, 018h, 018h, 000h, 000h
3924 00016CB5 007E0018180000      <1>
3925 00016CBC 00000000000076DC00- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h
3925 00016CC5 76DC0000000000      <1>
3926 00016CCC 00386C6C3800000000- <1>    db  000h, 038h, 06ch, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3926 00016CD5 000000000000000     <1>
3927 00016CDC 000000001818000000- <1>    db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3927 00016CE5 000000000000000     <1>
3928 00016CEC 000001800000000000- <1>    db  000h, 000h, 000h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 00fh, 00ch, 00ch, 00ch, 00ch
3928 00016CF5 00000F0C0C0C0C      <1>
3929 00016CFC 0CEC6C3C1C00000000- <1>    db  00ch, 0ech, 06ch, 03ch, 01ch, 000h, 000h, 000h, 000h, 0d8h, 06ch, 06ch, 06ch, 06ch, 06ch, 000h
3929 00016D05 D86C6C6C6C6C00      <1>
3930 00016D0C 0000000000000070D8- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 070h, 0d8h, 030h, 060h, 0c8h, 0f8h, 000h, 000h, 000h
3930 00016D15 3060C8F8000000      <1>
3931 00016D1C 000000000000007C- <1>      db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch, 07ch, 07ch, 07ch, 07ch, 07ch, 000h, 000h
3931 00016D25 7C7C7C7C7C0000      <1>
3932 00016D2C 000000000000000000- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3932 00016D35 000000000000000     <1>
3933                              <1> vgafont16:
3934 00016D3C 000000000000000000- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3934 00016D45 000000000000000     <1>
3935 00016D4C 00007E81A58181BD99- <1>    db  000h, 000h, 07eh, 081h, 0a5h, 081h, 081h, 0bdh, 099h, 081h, 081h, 07eh, 000h, 000h, 000h, 000h
3935 00016D55 81817E00000000      <1>
3936 00016D5C 00007EFFDBFFFFC3E7- <1>    db  000h, 000h, 07eh, 0ffh, 0dbh, 0ffh, 0ffh, 0c3h, 0e7h, 0ffh, 0ffh, 07eh, 000h, 000h, 000h, 000h
```

```
3936 00016D65 FFFF7E00000000      <1>
3937 00016D6C 000000006CFEFEFEFE- <1>     db  000h, 000h, 000h, 000h, 06ch, 0feh, 0feh, 0feh, 0feh, 07ch, 038h, 010h, 000h, 000h, 000h, 000h
3937 00016D75 7C381000000000      <1>
3938 00016D7C 0000000010387CFE7C- <1>     db  000h, 000h, 000h, 000h, 010h, 038h, 07ch, 0feh, 07ch, 038h, 010h, 000h, 000h, 000h, 000h, 000h
3938 00016D85 38100000000000      <1>
3939 00016D8C 000000183C3CE7E7E7- <1>     db  000h, 000h, 000h, 018h, 03ch, 03ch, 0e7h, 0e7h, 0e7h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
3939 00016D95 18183C00000000      <1>
3940 00016D9C 000000183C7EFFFF7E- <1>     db  000h, 000h, 000h, 018h, 03ch, 07eh, 0ffh, 0ffh, 07eh, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
3940 00016DA5 18183C00000000      <1>
3941 00016DAC 000000000000183C3C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 03ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h
3941 00016DB5 18000000000000      <1>
3942 00016DBC FFFFFFFFFFFFE7C3C3- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0e7h, 0c3h, 0c3h, 0e7h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
3942 00016DC5 E7FFFFFFFFFFFF      <1>
3943 00016DCC 00000000003C664242- <1>     db  000h, 000h, 000h, 000h, 000h, 03ch, 066h, 042h, 042h, 066h, 03ch, 000h, 000h, 000h, 000h, 000h
3943 00016DD5 663C0000000000      <1>
3944 00016DDC FFFFFFFFFFC399BDBD- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0c3h, 099h, 0bdh, 0bdh, 099h, 0c3h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
3944 00016DE5 99C3FFFFFFFFFF      <1>
3945 00016DEC 00001E0E1A3278CCCC- <1>     db  000h, 000h, 01eh, 00eh, 01ah, 032h, 078h, 0cch, 0cch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h
3945 00016DF5 CCCC7800000000      <1>
3946 00016DFC 00003C666666663C18- <1>     db  000h, 000h, 03ch, 066h, 066h, 066h, 066h, 03ch, 018h, 07eh, 018h, 018h, 000h, 000h, 000h, 000h
3946 00016E05 7E181800000000      <1>
3947 00016E0C 00003F333F30303030- <1>     db  000h, 000h, 03fh, 033h, 03fh, 030h, 030h, 030h, 030h, 070h, 0f0h, 0e0h, 000h, 000h, 000h, 000h
3947 00016E15 70F0E000000000      <1>
3948 00016E1C 00007F637F63636363- <1>     db  000h, 000h, 07fh, 063h, 07fh, 063h, 063h, 063h, 063h, 067h, 0e7h, 0e6h, 0c0h, 000h, 000h, 000h
3948 00016E25 67E7E6C0000000      <1>
3949 00016E2C 0000001818DB3CE73C- <1>     db  000h, 000h, 000h, 018h, 018h, 0dbh, 03ch, 0e7h, 03ch, 0dbh, 018h, 018h, 000h, 000h, 000h, 000h
3949 00016E35 DB181800000000      <1>
3950 00016E3C 0080C0E0F0F8FEF8F0- <1>     db  000h, 080h, 0c0h, 0e0h, 0f0h, 0f8h, 0feh, 0f8h, 0f0h, 0e0h, 0c0h, 080h, 000h, 000h, 000h, 000h
3950 00016E45 E0C08000000000      <1>
3951 00016E4C 0002060E1E3EFE3E1E- <1>     db  000h, 002h, 006h, 00eh, 01eh, 03eh, 0feh, 03eh, 01eh, 00eh, 006h, 002h, 000h, 000h, 000h, 000h
3951 00016E55 0E060200000000      <1>
3952 00016E5C 0000183C7E1818187E- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h, 000h
3952 00016E65 3C180000000000      <1>
3953 00016E6C 000066666666666666- <1>     db  000h, 000h, 066h, 066h, 066h, 066h, 066h, 066h, 066h, 000h, 066h, 066h, 000h, 000h, 000h, 000h
3953 00016E75 006666000000000     <1>
3954 00016E7C 00007FDBDBDB7B1B1B- <1>     db  000h, 000h, 07fh, 0dbh, 0dbh, 0dbh, 07bh, 01bh, 01bh, 01bh, 01bh, 01bh, 000h, 000h, 000h, 000h
3954 00016E85 1B1B1B00000000      <1>
3955 00016E8C 007CC660386CC6C66C- <1>     db  000h, 07ch, 0c6h, 060h, 038h, 06ch, 0c6h, 0c6h, 06ch, 038h, 00ch, 0c6h, 07ch, 000h, 000h, 000h
3955 00016E95 380CC67C000000      <1>
3956 00016E9C 00000000000000000FE- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 0feh, 0feh, 0feh, 000h, 000h, 000h, 000h
3956 00016EA5 FEFEFE00000000      <1>
3957 00016EAC 0000183C7E1818187E- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 07eh, 000h, 000h, 000h, 000h
3957 00016EB5 3C187E00000000      <1>
3958 00016EBC 0000183C7E18181818- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h
3958 00016EC5 18181800000000      <1>
3959 00016ECC 000018181818181818- <1>     db  000h, 000h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h
3959 00016ED5 7E3C1800000000      <1>
3960 00016EDC 0000000000180CFE0C- <1>     db  000h, 000h, 000h, 000h, 000h, 018h, 00ch, 0feh, 00ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h
3960 00016EE5 18000000000000      <1>
3961 00016EEC 00000000003060FE60- <1>     db  000h, 000h, 000h, 000h, 000h, 030h, 060h, 0feh, 060h, 030h, 000h, 000h, 000h, 000h, 000h, 000h
3961 00016EF5 30000000000000      <1>
3962 00016EFC 000000000000C0C0C0- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0c0h, 0c0h, 0c0h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h
3962 00016F05 FE0000000000000     <1>
3963 00016F0C 00000000002466FF66- <1>     db  000h, 000h, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h, 000h, 000h, 000h, 000h
3963 00016F15 24000000000000      <1>
3964 00016F1C 000000001038387C7C- <1>     db  000h, 000h, 000h, 000h, 010h, 038h, 038h, 07ch, 07ch, 0feh, 0feh, 000h, 000h, 000h, 000h, 000h
3964 00016F25 FEFE0000000000      <1>
3965 00016F2C 00000000FEFE7C7C38- <1>     db  000h, 000h, 000h, 000h, 0feh, 0feh, 07ch, 07ch, 038h, 038h, 010h, 000h, 000h, 000h, 000h, 000h
3965 00016F35 38100000000000      <1>
3966 00016F3C 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3966 00016F45 00000000000000      <1>
3967 00016F4C 0000183C3C3C181818- <1>     db  000h, 000h, 018h, 03ch, 03ch, 03ch, 018h, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h, 000h
3967 00016F55 001818000000000     <1>
3968 00016F5C 006666662400000000- <1>     db  000h, 066h, 066h, 066h, 024h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3968 00016F65 000000000000000     <1>
3969 00016F6C 0000006C6CFE6C6C6C- <1>     db  000h, 000h, 000h, 06ch, 06ch, 0feh, 06ch, 06ch, 06ch, 0feh, 06ch, 06ch, 000h, 000h, 000h, 000h
3969 00016F75 FE6C6C00000000      <1>
3970 00016F7C 18187CC6C2C07C0606- <1>     db  018h, 018h, 07ch, 0c6h, 0c2h, 0c0h, 07ch, 006h, 006h, 086h, 0c6h, 07ch, 018h, 018h, 000h, 000h
3970 00016F85 86C67C18180000      <1>
3971 00016F8C 00000000C2C60C1830- <1>     db  000h, 000h, 000h, 000h, 0c2h, 0c6h, 00ch, 018h, 030h, 060h, 0c6h, 086h, 000h, 000h, 000h, 000h
3971 00016F95 60C68600000000      <1>
3972 00016F9C 0000386C6C3876DCCC- <1>     db  000h, 000h, 038h, 06ch, 06ch, 038h, 076h, 0dch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
3972 00016FA5 CCCC7600000000      <1>
3973 00016FAC 003030306000000000- <1>     db  000h, 030h, 030h, 030h, 060h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3973 00016FB5 000000000000000     <1>
3974 00016FBC 00000C1830303030 30- <1>     db  000h, 000h, 00ch, 018h, 030h, 030h, 030h, 030h, 030h, 030h, 018h, 00ch, 000h, 000h, 000h, 000h
3974 00016FC5 30180C00000000      <1>
3975 00016FCC 000030180C0C0C0C0C- <1>     db  000h, 000h, 030h, 018h, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 018h, 030h, 000h, 000h, 000h, 000h
3975 00016FD5 0C183000000000      <1>
3976 00016FDC 0000000000663CFF3C- <1>     db  000h, 000h, 000h, 000h, 000h, 066h, 03ch, 0ffh, 03ch, 066h, 000h, 000h, 000h, 000h, 000h, 000h
3976 00016FE5 66000000000000      <1>
3977 00016FEC 000000000018187E18- <1>     db  000h, 000h, 000h, 000h, 000h, 018h, 018h, 07eh, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h
3977 00016FF5 18000000000000      <1>
3978 00016FFC 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 030h, 000h, 000h, 000h
3978 00017005 18181830000000      <1>
3979 0001700C 000000000000000FE00- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3979 00017015 000000000000000     <1>
3980 0001701C 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h
3980 00017025 001818000000000     <1>
3981 0001702C 0000000002060C1830- <1>     db  000h, 000h, 000h, 000h, 002h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 080h, 000h, 000h, 000h, 000h
3981 00017035 60C08000000000      <1>
3982 0001703C 00003C66C3C3DBDBC3- <1>     db  000h, 000h, 03ch, 066h, 0c3h, 0c3h, 0dbh, 0dbh, 0c3h, 0c3h, 066h, 03ch, 000h, 000h, 000h, 000h
3982 00017045 C3663C00000000      <1>
3983 0001704C 000018387818181818- <1>     db  000h, 000h, 018h, 038h, 078h, 018h, 018h, 018h, 018h, 018h, 018h, 07eh, 000h, 000h, 000h, 000h
3983 00017055 18187E00000000      <1>
3984 0001705C 00007CC6060C183060- <1>     db  000h, 000h, 07ch, 0c6h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 0c6h, 0feh, 000h, 000h, 000h, 000h
3984 00017065 C0C6FE00000000      <1>
3985 0001706C 00007CC606063C0606- <1>     db  000h, 000h, 07ch, 0c6h, 006h, 006h, 03ch, 006h, 006h, 006h, 0c6h, 07ch, 000h, 000h, 000h, 000h
3985 00017075 06C67C00000000      <1>
3986 0001707C 00000C1C3C6CCCFE0C- <1>     db  000h, 000h, 00ch, 01ch, 03ch, 06ch, 0cch, 0feh, 00ch, 00ch, 00ch, 01eh, 000h, 000h, 000h, 000h
3986 00017085 0C0C1E00000000      <1>
3987 0001708C 0000FEC0C0C0FC0606- <1>     db  000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 0fch, 006h, 006h, 006h, 0c6h, 07ch, 000h, 000h, 000h, 000h
3987 00017095 06C67C00000000      <1>
3988 0001709C 00003860C0C0FCC6C6- <1>     db  000h, 000h, 038h, 060h, 0c0h, 0c0h, 0fch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
3988 000170A5 C6C67C00000000      <1>
3989 000170AC 0000FEC606060C1830- <1>     db  000h, 000h, 0feh, 0c6h, 006h, 006h, 00ch, 018h, 030h, 030h, 030h, 030h, 000h, 000h, 000h, 000h
3989 000170B5 30303000000000      <1>
3990 000170BC 00007CC6C6C67CC6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
3990 000170C5 C6C67C00000000      <1>
3991 000170CC 00007CC6C6C67E0606- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 07eh, 006h, 006h, 006h, 00ch, 078h, 000h, 000h, 000h, 000h
3991 000170D5 060C7800000000      <1>
```

```
3992 000170DC 000000001818000000- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h
3992 000170E5 18180000000000      <1>
3993 000170EC 000000001818000000- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 018h, 018h, 030h, 000h, 000h, 000h, 000h
3993 000170F5 18183000000000      <1>
3994 000170FC 000000060C18306030- <1>     db  000h, 000h, 000h, 006h, 00ch, 018h, 030h, 060h, 030h, 018h, 00ch, 006h, 000h, 000h, 000h, 000h
3994 00017105 180C0600000000      <1>
3995 0001710C 00000000007E00007E- <1>     db  000h, 000h, 000h, 000h, 000h, 07eh, 000h, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 000h, 000h
3995 00017115 00000000000000      <1>
3996 0001711C 0000006030180C060C- <1>     db  000h, 000h, 000h, 060h, 030h, 018h, 00ch, 006h, 00ch, 018h, 030h, 060h, 000h, 000h, 000h, 000h
3996 00017125 18306000000000      <1>
3997 0001712C 00007CC6C60C181818- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 00ch, 018h, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h, 000h
3997 00017135 00181800000000      <1>
3998 0001713C 0000007CC6C6DEDEDE- <1>     db  000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0deh, 0deh, 0deh, 0dch, 0c0h, 07ch, 000h, 000h, 000h, 000h
3998 00017145 DCC07C00000000      <1>
3999 0001714C 000010386CC6C6FEC6- <1>     db  000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
3999 00017155 C6C6C600000000      <1>
4000 0001715C 0000FC6666667C6666- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 066h, 066h, 066h, 066h, 0fch, 000h, 000h, 000h, 000h
4000 00017165 6666FC00000000      <1>
4001 0001716C 00003C66C2C0C0C0C0- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0c0h, 0c0h, 0c2h, 066h, 03ch, 000h, 000h, 000h, 000h
4001 00017175 C2663C00000000      <1>
4002 0001717C 0000F86C6666666666- <1>     db  000h, 000h, 0f8h, 06ch, 066h, 066h, 066h, 066h, 066h, 066h, 06ch, 0f8h, 000h, 000h, 000h, 000h
4002 00017185 666CF800000000      <1>
4003 0001718C 0000FE666268786860- <1>     db  000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 060h, 062h, 066h, 0feh, 000h, 000h, 000h, 000h
4003 00017195 6266FE00000000      <1>
4004 0001719C 0000FE666268786860- <1>     db  000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h
4004 000171A5 6060F000000000      <1>
4005 000171AC 00003C66C2C0C0DEC6- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0deh, 0c6h, 0c6h, 066h, 03ah, 000h, 000h, 000h, 000h
4005 000171B5 C6663A00000000      <1>
4006 000171BC 0000C6C6C6C6FEC6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
4006 000171C5 C6C6C600000000      <1>
4007 000171CC 00003C181818181818- <1>     db  000h, 000h, 03ch, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
4007 000171D5 18183C00000000      <1>
4008 000171DC 00001E0C0C0C0C0CCC- <1>     db  000h, 000h, 01eh, 00ch, 00ch, 00ch, 00ch, 00ch, 0cch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h
4008 000171E5 CCCC7800000000      <1>
4009 000171EC 0000E666666C78786C- <1>     db  000h, 000h, 0e6h, 066h, 066h, 06ch, 078h, 078h, 06ch, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h
4009 000171F5 6666E600000000      <1>
4010 000171FC 0000F0606060606060- <1>     db  000h, 000h, 0f0h, 060h, 060h, 060h, 060h, 060h, 060h, 062h, 066h, 0feh, 000h, 000h, 000h, 000h
4010 00017205 6266FE00000000      <1>
4011 0001720C 0000C3E7FFFFDBC3C3- <1>     db  000h, 000h, 0c3h, 0e7h, 0ffh, 0ffh, 0dbh, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 000h, 000h, 000h, 000h
4011 00017215 C3C3C300000000      <1>
4012 0001721C 0000C6E6F6FEDECEC6- <1>     db  000h, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
4012 00017225 C6C6C600000000      <1>
4013 0001722C 00007CC6C6C6C6C6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4013 00017235 C6C67C00000000      <1>
4014 0001723C 0000FC6666667C6060- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 060h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h
4014 00017245 6060F000000000      <1>
4015 0001724C 00007CC6C6C6C6C6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0d6h, 0deh, 07ch, 00ch, 00eh, 000h, 000h
4015 00017255 D6DE7C0C0E0000      <1>
4016 0001725C 0000FC6666667C6C66- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 06ch, 066h, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h
4016 00017265 6666E600000000      <1>
4017 0001726C 00007CC6C660380C06- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 060h, 038h, 00ch, 006h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4017 00017275 C6C67C00000000      <1>
4018 0001727C 0000FFDB9918181818- <1>     db  000h, 000h, 0ffh, 0dbh, 099h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
4018 00017285 18183C00000000      <1>
4019 0001728C 0000C6C6C6C6C6C6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4019 00017295 C6C67C00000000      <1>
4020 0001729C 0000C3C3C3C3C3C3C3- <1>     db  000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h, 000h
4020 000172A5 663C1800000000      <1>
4021 000172AC 0000C3C3C3C3C3DBDB- <1>     db  000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 066h, 000h, 000h, 000h, 000h
4021 000172B5 FF666600000000      <1>
4022 000172BC 0000C3C3663C18183C- <1>     db  000h, 000h, 0c3h, 0c3h, 066h, 03ch, 018h, 018h, 03ch, 066h, 0c3h, 0c3h, 000h, 000h, 000h, 000h
4022 000172C5 66C3C300000000      <1>
4023 000172CC 0000C3C3C3663C1818- <1>     db  000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
4023 000172D5 18183C00000000      <1>
4024 000172DC 0000FFC3860C183060- <1>     db  000h, 000h, 0ffh, 0c3h, 086h, 00ch, 018h, 030h, 060h, 0c1h, 0c3h, 0ffh, 000h, 000h, 000h, 000h
4024 000172E5 C1C3FF00000000      <1>
4025 000172EC 00003C303030303030- <1>     db  000h, 000h, 03ch, 030h, 030h, 030h, 030h, 030h, 030h, 030h, 030h, 03ch, 000h, 000h, 000h, 000h
4025 000172F5 30303C00000000      <1>
4026 000172FC 00000080C0E070381C- <1>     db  000h, 000h, 000h, 080h, 0c0h, 0e0h, 070h, 038h, 01ch, 00eh, 006h, 002h, 000h, 000h, 000h, 000h
4026 00017305 0E060200000000      <1>
4027 0001730C 00003C0C0C0C0C0C0C- <1>     db  000h, 000h, 03ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 03ch, 000h, 000h, 000h, 000h
4027 00017315 0C0C3C00000000      <1>
4028 0001731C 10386CC600000000000- <1>     db  010h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4028 00017325 00000000000000      <1>
4029 0001732C 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 000h
4029 00017335 00000000FF0000      <1>
4030 0001733C 303018000000000000- <1>     db  030h, 030h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4030 00017345 00000000000000      <1>
4031 0001734C 0000000000780C7CCC- <1>     db  000h, 000h, 000h, 000h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4031 00017355 CCCC7600000000      <1>
4032 0001735C 0000E06060786C6666- <1>     db  000h, 000h, 0e0h, 060h, 060h, 078h, 06ch, 066h, 066h, 066h, 066h, 07ch, 000h, 000h, 000h, 000h
4032 00017365 66667C00000000      <1>
4033 0001736C 00000000007CC6C0C0- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c0h, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4033 00017375 C0C67C00000000      <1>
4034 0001737C 00001C0C0C3C6CCCCC- <1>     db  000h, 000h, 01ch, 00ch, 00ch, 03ch, 06ch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4034 00017385 CCCC7600000000      <1>
4035 0001738C 00000000007CC6FEC0- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4035 00017395 C0C67C00000000      <1>
4036 0001739C 0000386C6460F06060- <1>     db  000h, 000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h
4036 000173A5 6060F000000000      <1>
4037 000173AC 000000000076CCCCCC- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 0cch, 0cch, 07ch, 00ch, 0cch, 078h, 000h
4037 000173B5 CCCC7C0CCC7800      <1>
4038 000173BC 0000E060606C766666- <1>     db  000h, 000h, 0e0h, 060h, 060h, 06ch, 076h, 066h, 066h, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h
4038 000173C5 6666E600000000      <1>
4039 000173CC 000018180038181818- <1>     db  000h, 000h, 018h, 018h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
4039 000173D5 18183C00000000      <1>
4040 000173DC 00000606000E060606- <1>     db  000h, 000h, 006h, 006h, 000h, 00eh, 006h, 006h, 006h, 006h, 006h, 006h, 066h, 066h, 03ch, 000h
4040 000173E5 06060666663C00      <1>
4041 000173EC 0000E06060666C7878- <1>     db  000h, 000h, 0e0h, 060h, 060h, 066h, 06ch, 078h, 078h, 06ch, 066h, 0e6h, 000h, 000h, 000h, 000h
4041 000173F5 6C66E600000000      <1>
4042 000173FC 000038181818181818- <1>     db  000h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
4042 00017405 18183C00000000      <1>
4043 0001740C 0000000000E6FFDBDB- <1>     db  000h, 000h, 000h, 000h, 000h, 0e6h, 0ffh, 0dbh, 0dbh, 0dbh, 0dbh, 0dbh, 000h, 000h, 000h, 000h
4043 00017415 DBDBDB00000000      <1>
4044 0001741C 0000000000DC666666- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 066h, 000h, 000h, 000h, 000h
4044 00017425 66666600000000      <1>
4045 0001742C 00000000007CC6C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4045 00017435 C6C67C00000000      <1>
4046 0001743C 0000000000DC666666- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 07ch, 060h, 060h, 0f0h, 000h
4046 00017445 66667C6060F000      <1>
4047 0001744C 000000000076CCCCCC- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 0cch, 0cch, 07ch, 00ch, 00ch, 01eh, 000h
```

```
4047 00017455 CCCC7C0C0C1E00      <1>
4048 0001745C 0000000000DC766660- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 076h, 066h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h
4048 00017465 6060F000000000      <1>
4049 0001746C 00000000007CC66038- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 060h, 038h, 00ch, 0c6h, 07ch, 000h, 000h, 000h, 000h
4049 00017475 0CC67C00000000      <1>
4050 0001747C 0000103030FC303030- <1>     db  000h, 000h, 010h, 030h, 030h, 0fch, 030h, 030h, 030h, 030h, 036h, 01ch, 000h, 000h, 000h, 000h
4050 00017485 30361C00000000      <1>
4051 0001748C 0000000000CCCCCCCC- <1>     db  000h, 000h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4051 00017495 CCCC7600000000      <1>
4052 0001749C 0000000000C3C3C3C3- <1>     db  000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h, 000h
4052 000174A5 663C1800000000      <1>
4053 000174AC 0000000000C3C3C3DB- <1>     db  000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 000h, 000h, 000h, 000h
4053 000174B5 DBFF6600000000      <1>
4054 000174BC 0000000000C3663C18- <1>     db  000h, 000h, 000h, 000h, 000h, 0c3h, 066h, 03ch, 018h, 03ch, 066h, 0c3h, 000h, 000h, 000h, 000h
4054 000174C5 3C66C300000000      <1>
4055 000174CC 0000000000C6C6C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 0f8h, 000h
4055 000174D5 C6C67E060CF800      <1>
4056 000174DC 0000000000FECC1830- <1>     db  000h, 000h, 000h, 000h, 000h, 0feh, 0cch, 018h, 030h, 060h, 0c6h, 0feh, 000h, 000h, 000h, 000h
4056 000174E5 60C6FE00000000      <1>
4057 000174EC 00000E181818701818- <1>     db  000h, 000h, 00eh, 018h, 018h, 018h, 070h, 018h, 018h, 018h, 018h, 00eh, 000h, 000h, 000h, 000h
4057 000174F5 18180E00000000      <1>
4058 000174FC 0000181818180018 18- <1>     db  000h, 000h, 018h, 018h, 018h, 018h, 000h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h
4058 00017505 18181800000000      <1>
4059 0001750C 0000701818180E1818- <1>     db  000h, 000h, 070h, 018h, 018h, 018h, 00eh, 018h, 018h, 018h, 018h, 070h, 000h, 000h, 000h, 000h
4059 00017515 18187000000000      <1>
4060 0001751C 000076DC0000000000- <1>     db  000h, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4060 00017525 00000000000000      <1>
4061 0001752C 0000000010386CC6C6- <1>     db  000h, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h
4061 00017535 C6FE0000000000      <1>
4062 0001753C 00003C66C2C0C0C0C2- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0c0h, 0c2h, 066h, 03ch, 00ch, 006h, 07ch, 000h, 000h
4062 00017545 663C0C067C0000      <1>
4063 0001754C 0000CC0000CCCCCCCC- <1>     db  000h, 000h, 0cch, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4063 00017555 CCCC7600000000      <1>
4064 0001755C 000C1830007CC6FEC0- <1>     db  000h, 00ch, 018h, 030h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4064 00017565 C0C67C00000000      <1>
4065 0001756C 0010386C00780C7CCC- <1>     db  000h, 010h, 038h, 06ch, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4065 00017575 CCCC7600000000      <1>
4066 0001757C 0000CC0000780C7CCC- <1>     db  000h, 000h, 0cch, 000h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4066 00017585 CCCC7600000000      <1>
4067 0001758C 0060301800780C7CCC- <1>     db  000h, 060h, 030h, 018h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4067 00017595 CCCC7600000000      <1>
4068 0001759C 00386C3800780C7CCC- <1>     db  000h, 038h, 06ch, 038h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4068 000175A5 CCCC7600000000      <1>
4069 000175AC 000000003C66606066- <1>     db  000h, 000h, 000h, 000h, 03ch, 066h, 060h, 060h, 066h, 03ch, 00ch, 006h, 03ch, 000h, 000h, 000h
4069 000175B5 3C0C063C000000      <1>
4070 000175BC 0010386C007CC6FEC0- <1>     db  000h, 010h, 038h, 06ch, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4070 000175C5 C0C67C00000000      <1>
4071 000175CC 0000C600007CC6FEC0- <1>     db  000h, 000h, 0c6h, 000h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4071 000175D5 C0C67C00000000      <1>
4072 000175DC 00603018007CC6FEC0- <1>     db  000h, 060h, 030h, 018h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4072 000175E5 C0C67C00000000      <1>
4073 000175EC 000066000038181818- <1>     db  000h, 000h, 066h, 000h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
4073 000175F5 18183C00000000      <1>
4074 000175FC 00183C660038181818- <1>     db  000h, 018h, 03ch, 066h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
4074 00017605 18183C00000000      <1>
4075 0001760C 006030180038181818- <1>     db  000h, 060h, 030h, 018h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
4075 00017615 18183C00000000      <1>
4076 0001761C 00C60010386CC6C6FE- <1>     db  000h, 0c6h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
4076 00017625 C6C6C600000000      <1>
4077 0001762C 386C3800386CC6C6FE- <1>     db  038h, 06ch, 038h, 000h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
4077 00017635 C6C6C600000000      <1>
4078 0001763C 18306000FE66607C60- <1>     db  018h, 030h, 060h, 000h, 0feh, 066h, 060h, 07ch, 060h, 060h, 066h, 0feh, 000h, 000h, 000h, 000h
4078 00017645 6066FE00000000      <1>
4079 0001764C 00000000006E3B1B7E- <1>     db  000h, 000h, 000h, 000h, 000h, 06eh, 03bh, 01bh, 07eh, 0d8h, 0dch, 077h, 000h, 000h, 000h, 000h
4079 00017655 D8DC7700000000      <1>
4080 0001765C 00003E6CCCCCFECCCC- <1>     db  000h, 000h, 03eh, 06ch, 0cch, 0cch, 0feh, 0cch, 0cch, 0cch, 0cch, 0ceh, 000h, 000h, 000h, 000h
4080 00017665 CCCCCE00000000      <1>
4081 0001766C 0010386C007CC6C6C6- <1>     db  000h, 010h, 038h, 06ch, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4081 00017675 C6C67C00000000      <1>
4082 0001767C 0000C600007CC6C6C6- <1>     db  000h, 000h, 0c6h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4082 00017685 C6C67C00000000      <1>
4083 0001768C 00603018007CC6C6C6- <1>     db  000h, 060h, 030h, 018h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4083 00017695 C6C67C00000000      <1>
4084 0001769C 003078CC00CCCCCCCC- <1>     db  000h, 030h, 078h, 0cch, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4084 000176A5 CCCC7600000000      <1>
4085 000176AC 0060301800CCCCCCCC- <1>     db  000h, 060h, 030h, 018h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4085 000176B5 CCCC7600000000      <1>
4086 000176BC 0000C60000C6C6C6C6- <1>     db  000h, 000h, 0c6h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 078h, 000h
4086 000176C5 C6C67E060C7800      <1>
4087 000176CC 00C6007CC6C6C6C6C6- <1>     db  000h, 0c6h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4087 000176D5 C6C67C00000000      <1>
4088 000176DC 00C600C6C6C6C6C6C6- <1>     db  000h, 0c6h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4088 000176E5 C6C67C00000000      <1>
4089 000176EC 0018187EC3C0C0C0C3- <1>     db  000h, 018h, 018h, 07eh, 0c3h, 0c0h, 0c0h, 0c0h, 0c3h, 07eh, 018h, 018h, 000h, 000h, 000h, 000h
4089 000176F5 7E181800000000      <1>
4090 000176FC 00386C6460F0606060- <1>     db  000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h, 060h, 060h, 0e6h, 0fch, 000h, 000h, 000h, 000h
4090 00017705 60E6FC00000000      <1>
4091 0001770C 0000C3663C18FF18FF- <1>     db  000h, 000h, 0c3h, 066h, 03ch, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 000h, 000h, 000h, 000h
4091 00017715 18181800000000      <1>
4092 0001771C 00FC66667C62666F66- <1>     db  000h, 0fch, 066h, 066h, 07ch, 062h, 066h, 06fh, 066h, 066h, 066h, 0f3h, 000h, 000h, 000h, 000h
4092 00017725 6666F300000000      <1>
4093 0001772C 000E1B1818187E1818- <1>     db  000h, 00eh, 01bh, 018h, 018h, 018h, 07eh, 018h, 018h, 018h, 018h, 018h, 0d8h, 070h, 000h, 000h
4093 00017735 181818D8700000      <1>
4094 0001773C 0018306000780C7CCC- <1>     db  000h, 018h, 030h, 060h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4094 00017745 CCCC7600000000      <1>
4095 0001774C 000C18300038181818- <1>     db  000h, 00ch, 018h, 030h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
4095 00017755 18183C00000000      <1>
4096 0001775C 00183060007CC6C6C6- <1>     db  000h, 018h, 030h, 060h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4096 00017765 C6C67C00000000      <1>
4097 0001776C 0018306000CCCCCCCC- <1>     db  000h, 018h, 030h, 060h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
4097 00017775 CCCC7600000000      <1>
4098 0001777C 000076DC00DC666666- <1>     db  000h, 000h, 076h, 0dch, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 066h, 000h, 000h, 000h, 000h
4098 00017785 66666600000000      <1>
4099 0001778C 76DC00C6E6F6FEDECE- <1>     db  076h, 0dch, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
4099 00017795 C6C6C600000000      <1>
4100 0001779C 003C6C6C3E007E0000- <1>     db  000h, 03ch, 06ch, 06ch, 03eh, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4100 000177A5 00000000000000      <1>
4101 000177AC 00386C6C38007C0000- <1>     db  000h, 038h, 06ch, 06ch, 038h, 000h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4101 000177B5 00000000000000      <1>
4102 000177BC 0000303000303060C0- <1>     db  000h, 000h, 030h, 030h, 000h, 030h, 030h, 060h, 0c0h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
4102 000177C5 C6C67C00000000      <1>
```

```
4103 000177CC 000000000000FEC0C0- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h, 000h
4103 000177D5 C0C00000000000      <1>
4104 000177DC 000000000000FE0606- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0feh, 006h, 006h, 006h, 006h, 000h, 000h, 000h, 000h, 000h
4104 000177E5 06060000000000      <1>
4105 000177EC 00C0C0C2C6CC183060- <1>     db  000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 060h, 0ceh, 09bh, 006h, 00ch, 01fh, 000h, 000h
4105 000177F5 CE9B060C1F0000      <1>
4106 000177FC 00C0C0C2C6CC183066- <1>     db  000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 066h, 0ceh, 096h, 03eh, 006h, 006h, 000h, 000h
4106 00017805 CE963E06060000      <1>
4107 0001780C 000018180018183C- <1>     db  000h, 000h, 018h, 018h, 000h, 018h, 018h, 018h, 03ch, 03ch, 03ch, 018h, 000h, 000h, 000h, 000h
4107 00017815 3C3C1800000000      <1>
4108 0001781C 0000000000366CD86C- <1>     db  000h, 000h, 000h, 000h, 000h, 036h, 06ch, 0d8h, 06ch, 036h, 000h, 000h, 000h, 000h, 000h, 000h
4108 00017825 36000000000000      <1>
4109 0001782C 0000000000D86C366C- <1>     db  000h, 000h, 000h, 000h, 000h, 0d8h, 06ch, 036h, 06ch, 0d8h, 000h, 000h, 000h, 000h, 000h, 000h
4109 00017835 D8000000000000      <1>
4110 0001783C 1144114411441144411- <1>     db  011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h
4110 00017845 44114411441144      <1>
4111 0001784C 55AA55AA55AA55AA55- <1>     db  055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah
4111 00017855 AA55AA55AA55AA      <1>
4112 0001785C DD77DD77DD77DD77DD- <1>     db  0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h
4112 00017865 77DD77DD77DD77      <1>
4113 0001786C 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4113 00017875 18181818181818      <1>
4114 0001787C 18181818181818F818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4114 00017885 18181818181818      <1>
4115 0001788C 1818181818F818F818- <1>     db  018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4115 00017895 18181818181818      <1>
4116 0001789C 363636363636F636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4116 000178A5 36363636363636      <1>
4117 000178AC 000000000000FE36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4117 000178B5 36363636363636      <1>
4118 000178BC 0000000000F818F818- <1>     db  000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4118 000178C5 18181818181818      <1>
4119 000178CC 3636363636F606F636- <1>     db  036h, 036h, 036h, 036h, 036h, 0f6h, 006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4119 000178D5 36363636363636      <1>
4120 000178DC 363636363636363636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4120 000178E5 36363636363636      <1>
4121 000178EC 0000000000FE06F636- <1>     db  000h, 000h, 000h, 000h, 000h, 0feh, 006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4121 000178F5 36363636363636      <1>
4122 000178FC 3636363636F606FE00- <1>     db  036h, 036h, 036h, 036h, 036h, 0f6h, 006h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4122 00017905 00000000000000      <1>
4123 0001790C 363636363636FE00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4123 00017915 00000000000000      <1>
4124 0001791C 1818181818F818F800- <1>     db  018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4124 00017925 00000000000000      <1>
4125 0001792C 000000000000000F818- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4125 00017935 18181818181818      <1>
4126 0001793C 181818181818181F00- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4126 00017945 00000000000000      <1>
4127 0001794C 181818181818FF00- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4127 00017955 00000000000000      <1>
4128 0001795C 000000000000000FF18- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4128 00017965 18181818181818      <1>
4129 0001796C 181818181818181F18- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4129 00017975 18181818181818      <1>
4130 0001797C 000000000000000FF00- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4130 00017985 00000000000000      <1>
4131 0001798C 181818181818FF18- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4131 00017995 18181818181818      <1>
4132 0001799C 18181818181F181F18- <1>     db  018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4132 000179A5 18181818181818      <1>
4133 000179AC 363636363636363736- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4133 000179B5 36363636363636      <1>
4134 000179BC 363636363637303F00- <1>     db  036h, 036h, 036h, 036h, 036h, 037h, 030h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4134 000179C5 00000000000000      <1>
4135 000179CC 00000000003F303736- <1>     db  000h, 000h, 000h, 000h, 000h, 03fh, 030h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4135 000179D5 36363636363636      <1>
4136 000179DC 3636363636F700FF00- <1>     db  036h, 036h, 036h, 036h, 036h, 0f7h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4136 000179E5 00000000000000      <1>
4137 000179EC 0000000000FF00F736- <1>     db  000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0f7h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4137 000179F5 36363636363636      <1>
4138 000179FC 363636363637303736- <1>     db  036h, 036h, 036h, 036h, 036h, 037h, 030h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4138 00017A05 36363636363636      <1>
4139 00017A0C 0000000000FF00FF00- <1>     db  000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4139 00017A15 00000000000000      <1>
4140 00017A1C 3636363636F700F736- <1>     db  036h, 036h, 036h, 036h, 036h, 0f7h, 000h, 0f7h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4140 00017A25 36363636363636      <1>
4141 00017A2C 1818181818FF00FF00- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4141 00017A35 00000000000000      <1>
4142 00017A3C 36363636363636FF00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4142 00017A45 00000000000000      <1>
4143 00017A4C 0000000000FF00FF18- <1>     db  000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4143 00017A55 18181818181818      <1>
4144 00017A5C 000000000000000FF36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4144 00017A65 36363636363636      <1>
4145 00017A6C 363636363636363F00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4145 00017A75 00000000000000      <1>
4146 00017A7C 18181818181F181F00- <1>     db  018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4146 00017A85 00000000000000      <1>
4147 00017A8C 00000000001F181F18- <1>     db  000h, 000h, 000h, 000h, 000h, 01fh, 018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4147 00017A95 18181818181818      <1>
4148 00017A9C 000000000000003F36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 03fh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4148 00017AA5 36363636363636      <1>
4149 00017AAC 363636363636363FF36- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
4149 00017AB5 36363636363636      <1>
4150 00017ABC 1818181818FF18FF18- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4150 00017AC5 18181818181818      <1>
4151 00017ACC 18181818181818F800- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4151 00017AD5 00000000000000      <1>
4152 00017ADC 000000000000001F18- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4152 00017AE5 18181818181818      <1>
4153 00017AEC FFFFFFFFFFFFFFFFFF- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
4153 00017AF5 FFFFFFFFFFFFFF      <1>
4154 00017AFC 000000000000000FFFF- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
4154 00017B05 FFFFFFFFFFFFFF      <1>
4155 00017B0C F0F0F0F0F0F0F0F0F0- <1>     db  0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h
4155 00017B15 F0F0F0F0F0F0F0      <1>
4156 00017B1C 0F0F0F0F0F0F0F0F0F- <1>     db  00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh
4156 00017B25 0F0F0F0F0F0F0F      <1>
4157 00017B2C FFFFFFFFFFFFFF0000- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4157 00017B35 00000000000000      <1>
4158 00017B3C 000000000076DCD8D8- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0dch, 0d8h, 0d8h, 0d8h, 0dch, 076h, 000h, 000h, 000h, 000h
```

```
4158 00017B45 D8DC7600000000      <1>
4159 00017B4C 000078CCCCCCD8CCC6- <1>     db  000h, 000h, 078h, 0cch, 0cch, 0cch, 0d8h, 0cch, 0c6h, 0c6h, 0c6h, 0cch, 000h, 000h, 000h, 000h
4159 00017B55 C6C6CC00000000      <1>
4160 00017B5C 0000FEC6C6C0C0C0C0- <1>     db  000h, 000h, 0feh, 0c6h, 0c6h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h
4160 00017B65 C0C0C000000000      <1>
4161 00017B6C 00000000FE6C6C6C6C- <1>     db  000h, 000h, 000h, 000h, 0feh, 06ch, 06ch, 06ch, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 000h
4161 00017B75 6C6C6C00000000      <1>
4162 00017B7C 000000FEC660301830- <1>     db  000h, 000h, 000h, 0feh, 0c6h, 060h, 030h, 018h, 030h, 060h, 0c6h, 0feh, 000h, 000h, 000h, 000h
4162 00017B85 60C6FE00000000      <1>
4163 00017B8C 00000000007ED8D8D8- <1>     db  000h, 000h, 000h, 000h, 000h, 07eh, 0d8h, 0d8h, 0d8h, 0d8h, 0d8h, 070h, 000h, 000h, 000h, 000h
4163 00017B95 D8D87000000000      <1>
4164 00017B9C 000000066666666666- <1>     db  000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h, 066h, 07ch, 060h, 060h, 0c0h, 000h, 000h, 000h
4164 00017BA5 7C6060C0000000      <1>
4165 00017BAC 0000000076DC181818- <1>     db  000h, 000h, 000h, 000h, 076h, 0dch, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h
4165 00017BB5 18181800000000      <1>
4166 00017BBC 0000007E183C666666- <1>     db  000h, 000h, 000h, 07eh, 018h, 03ch, 066h, 066h, 066h, 03ch, 018h, 07eh, 000h, 000h, 000h, 000h
4166 00017BC5 3C187E00000000      <1>
4167 00017BCC 000000386CC6C6FEC6- <1>     db  000h, 000h, 000h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h, 000h
4167 00017BD5 C66C3800000000      <1>
4168 00017BDC 0000386CC6C6C66C6C- <1>     db  000h, 000h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 06ch, 06ch, 06ch, 06ch, 0eeh, 000h, 000h, 000h, 000h
4168 00017BE5 6C6CEE00000000      <1>
4169 00017BEC 00001E30180C3E6666- <1>     db  000h, 000h, 01eh, 030h, 018h, 00ch, 03eh, 066h, 066h, 066h, 066h, 03ch, 000h, 000h, 000h, 000h
4169 00017BF5 66663C00000000      <1>
4170 00017BFC 00000000007EDBDBDB- <1>     db  000h, 000h, 000h, 000h, 000h, 07eh, 0dbh, 0dbh, 0dbh, 07eh, 000h, 000h, 000h, 000h, 000h, 000h
4170 00017C05 7E000000000000      <1>
4171 00017C0C 00000003067EDBDBF3- <1>     db  000h, 000h, 000h, 003h, 006h, 07eh, 0dbh, 0dbh, 0f3h, 07eh, 060h, 0c0h, 000h, 000h, 000h, 000h
4171 00017C15 7E60C000000000      <1>
4172 00017C1C 00001C3060607C6060- <1>     db  000h, 000h, 01ch, 030h, 060h, 060h, 07ch, 060h, 060h, 060h, 030h, 01ch, 000h, 000h, 000h, 000h
4172 00017C25 60301C00000000      <1>
4173 00017C2C 0000007CC6C6C6C6C6- <1>     db  000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
4173 00017C35 C6C6C600000000      <1>
4174 00017C3C 00000000FE0000FE00- <1>     db  000h, 000h, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 000h, 000h, 000h
4174 00017C45 00FE0000000000      <1>
4175 00017C4C 0000000018187E1818- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 07eh, 018h, 018h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h
4175 00017C55 0000FF00000000      <1>
4176 00017C5C 00000030180C060C18- <1>     db  000h, 000h, 000h, 030h, 018h, 00ch, 006h, 00ch, 018h, 030h, 000h, 07eh, 000h, 000h, 000h, 000h
4176 00017C65 30007E00000000      <1>
4177 00017C6C 0000000C1830603018- <1>     db  000h, 000h, 000h, 00ch, 018h, 030h, 060h, 030h, 018h, 00ch, 000h, 07eh, 000h, 000h, 000h, 000h
4177 00017C75 0C007E00000000      <1>
4178 00017C7C 00000E1B1B18181818- <1>     db  000h, 000h, 00eh, 01bh, 01bh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
4178 00017C85 18181818181818      <1>
4179 00017C8C 18181818181818D8- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0d8h, 0d8h, 0d8h, 070h, 000h, 000h, 000h, 000h
4179 00017C95 D8D87000000000      <1>
4180 00017C9C 000000001818007E00- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 07eh, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h
4180 00017CA5 18180000000000      <1>
4181 00017CAC 000000000076DC0076- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h
4181 00017CB5 DC0000000000000     <1>
4182 00017CBC 00386C6C3800000000- <1>     db  000h, 038h, 06ch, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4182 00017CC5 00000000000000      <1>
4183 00017CCC 000000000000001818- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4183 00017CD5 00000000000000      <1>
4184 00017CDC 000000000000000018- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4184 00017CE5 00000000000000      <1>
4185 00017CEC 000F0C0C0C0C0CEC6C- <1>     db  000h, 00fh, 00ch, 00ch, 00ch, 00ch, 00ch, 0ech, 06ch, 06ch, 03ch, 01ch, 000h, 000h, 000h, 000h
4185 00017CF5 6C3C1C00000000      <1>
4186 00017CFC 00D86C6C6C6C6C0000- <1>     db  000h, 0d8h, 06ch, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4186 00017D05 00000000000000      <1>
4187 00017D0C 0070D83060C8F80000- <1>     db  000h, 070h, 0d8h, 030h, 060h, 0c8h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4187 00017D15 00000000000000      <1>
4188 00017D1C 000000007C7C7C7C7C- <1>     db  000h, 000h, 000h, 000h, 07ch, 07ch, 07ch, 07ch, 07ch, 07ch, 07ch, 000h, 000h, 000h, 000h, 000h
4188 00017D25 7C7C0000000000      <1>
4189 00017D2C 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
4189 00017D35 00000000000000      <1>
4190                              <1>
4191                              <1> ; 01/01/2021 (TRDOS 386 v2.0.3)
4192                              <1>
4193                              <1> %if 0
4194                              <1>
4195                              <1> vgafont14alt:
4196                              <1>     db  01dh, 000h, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h, 000h, 000h, 000h, 022h
4197                              <1>     db  000h, 063h, 063h, 063h, 022h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 02bh, 000h
4198                              <1>     db  000h, 000h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 02dh, 000h, 000h
4199                              <1>     db  000h, 000h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 04dh, 000h, 000h, 0c3h
4200                              <1>     db  0e7h, 0ffh, 0dbh, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 000h, 000h, 000h, 054h, 000h, 000h, 0ffh, 0dbh
4201                              <1>     db  099h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 056h, 000h, 000h, 0c3h, 0c3h, 0c3h
4202                              <1>     db  0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h, 057h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h
4203                              <1>     db  0dbh, 0dbh, 0ffh, 066h, 066h, 000h, 000h, 000h, 058h, 000h, 000h, 0c3h, 0c3h, 066h, 03ch, 018h
4204                              <1>     db  03ch, 066h, 0c3h, 0c3h, 000h, 000h, 000h, 059h, 000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h
4205                              <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 05ah, 000h, 000h, 0ffh, 0c3h, 086h, 00ch, 018h, 030h, 061h
4206                              <1>     db  0c3h, 0ffh, 000h, 000h, 000h, 06dh, 000h, 000h, 000h, 000h, 000h, 0e6h, 0ffh, 0dbh, 0dbh, 0dbh
4207                              <1>     db  0dbh, 000h, 000h, 000h, 076h, 000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h
4208                              <1>     db  000h, 000h, 000h, 077h, 000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 000h
4209                              <1>     db  000h, 000h, 091h, 000h, 000h, 000h, 000h, 06eh, 03bh, 01bh, 07eh, 0d8h, 0dch, 077h, 000h, 000h
4210                              <1>     db  000h, 09bh, 000h, 018h, 018h, 07eh, 0c3h, 0c0h, 0c0h, 0c3h, 07eh, 018h, 018h, 000h, 000h, 000h
4211                              <1>     db  09dh, 000h, 000h, 0c3h, 066h, 03ch, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 000h, 000h, 000h, 09eh
4212                              <1>     db  000h, 0fch, 066h, 066h, 07ch, 062h, 066h, 06fh, 066h, 066h, 0f3h, 000h, 000h, 000h, 0f1h, 000h
4213                              <1>     db  000h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h, 000h, 0ffh, 000h, 000h, 000h, 0f6h, 000h, 000h
4214                              <1>     db  018h, 018h, 000h, 000h, 0ffh, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h
4215                              <1> vgafont16alt:
4216                              <1>     db  01dh, 000h, 000h, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h, 000h, 000h, 000h
4217                              <1>     db  000h, 030h, 000h, 000h, 03ch, 066h, 0c3h, 0c3h, 0dbh, 0dbh, 0c3h, 0c3h, 066h, 03ch, 000h, 000h
4218                              <1>     db  000h, 000h, 04dh, 000h, 000h, 0c3h, 0e7h, 0ffh, 0ffh, 0dbh, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 000h
4219                              <1>     db  000h, 000h, 000h, 054h, 000h, 000h, 0ffh, 0dbh, 099h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch
4220                              <1>     db  000h, 000h, 000h, 000h, 056h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch
4221                              <1>     db  018h, 000h, 000h, 000h, 000h, 057h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh
4222                              <1>     db  066h, 066h, 000h, 000h, 000h, 000h, 058h, 000h, 000h, 0c3h, 0c3h, 066h, 03ch, 018h, 018h, 03ch
4223                              <1>     db  066h, 0c3h, 0c3h, 000h, 000h, 000h, 000h, 059h, 000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h
4224                              <1>     db  018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 05ah, 000h, 000h, 0ffh, 0c3h, 086h, 00ch, 018h
4225                              <1>     db  030h, 060h, 0c1h, 0c3h, 0ffh, 000h, 000h, 000h, 000h, 06dh, 000h, 000h, 000h, 000h, 000h, 0e6h
4226                              <1>     db  0ffh, 0dbh, 0dbh, 0dbh, 0dbh, 0dbh, 000h, 000h, 000h, 000h, 076h, 000h, 000h, 000h, 000h, 000h
4227                              <1>     db  0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h, 000h, 077h, 000h, 000h, 000h, 000h
4228                              <1>     db  000h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 000h, 000h, 000h, 000h, 078h, 000h, 000h, 000h
4229                              <1>     db  000h, 000h, 0c3h, 066h, 03ch, 018h, 03ch, 066h, 0c3h, 000h, 000h, 000h, 000h, 091h, 000h, 000h
4230                              <1>     db  000h, 000h, 000h, 06eh, 03bh, 01bh, 07eh, 0d8h, 0dch, 077h, 000h, 000h, 000h, 000h, 09bh, 000h
4231                              <1>     db  018h, 018h, 07eh, 0c3h, 0c0h, 0c0h, 0c0h, 0c3h, 07eh, 018h, 018h, 000h, 000h, 000h, 000h, 09dh
4232                              <1>     db  000h, 000h, 0c3h, 066h, 03ch, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 000h, 000h, 000h, 000h
4233                              <1>     db  09eh, 000h, 0fch, 066h, 066h, 07ch, 062h, 066h, 06fh, 066h, 066h, 066h, 0f3h, 000h, 000h, 000h
4234                              <1>     db  000h, 0abh, 000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 060h, 0ceh, 09bh, 006h, 00ch, 01fh
4235                              <1>     db  000h, 000h, 0ach, 000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 066h, 0ceh, 096h, 03eh, 006h
4236                              <1>     db  006h, 000h, 000h, 000h
4237                              <1>
```

```
4238                            <1> %endif
3456
3457                                    ; 20/11/2020
3458                                    vbe2_bochs_vbios:
3459                                    ;       db "BOCHS/QEMU"
3460 00017D3C 424F4348532F51454D-               db "BOCHS/QEMU/VIRTUALBOX"  ; 26/11/2020
3460 00017D45 552F5649525455414C-
3460 00017D4E 424F58
3461                                    ;vbe_vnumber equ vbe2_bochs_vbios + 28 ; "3" or "2"
3462                                    ; 26/11/2020
3463                                    vbe_vnumber equ vbe2_bochs_vbios + 30 ; "3" or "2"
3464                                    ; 15/11/2020
3465                                    vesa_vbe3_bios_msg:
3466                                            ;db " VESA VBE version 3 Video BIOS ..."
3467 00017D51 205645534120564245-               db " VESA VBE3 Video BIOS ..." ; 26/11/2020
3467 00017D5A 3320566964656F2042-
3467 00017D63 494F53202E2E2E
3468 00017D6A 0D0A0D0A00                        db 0Dh, 0Ah, 0Dh, 0Ah, 0
3469
3470                                    ; 28/02/2021
3471 00017D6F 18                        truecolor: db 24
3472
3473                                    align 2
3474
3475                                    ; EPOCH Variables
3476                                    ; 13/04/2015 - Retro UNIX 386 v1 Beginning
3477                                    ; 09/04/2013 epoch variables
3478                                    ; Retro UNIX 8086 v1 Prototype: UNIXCOPY.ASM, 10/03/2013
3479                                    ;
3480 00017D70 B207                     year:       dw 1970
3481 00017D72 0100                     month:      dw 1
3482 00017D74 0100                     day:  dw 1
3483 00017D76 0000                     hour:       dw 0
3484 00017D78 0000                     minute: dw 0
3485 00017D7A 0000                     second: dw 0
3486
3487                                    DMonth:
3488 00017D7C 0000                              dw 0
3489 00017D7E 1F00                              dw 31
3490 00017D80 3B00                              dw 59
3491 00017D82 5A00                              dw 90
3492 00017D84 7800                              dw 120
3493 00017D86 9700                              dw 151
3494 00017D88 B500                              dw 181
3495 00017D8A D400                              dw 212
3496 00017D8C F300                              dw 243
3497 00017D8E 1101                              dw 273
3498 00017D90 3001                              dw 304
3499 00017D92 4E01                              dw 334
3500
3501                                    ; 15/11/2020
3502 00017D94 00                        db    0
3503                                    kernel_version_msg: ; 17/04/2021
3504 00017D95 5452444F5320283338-       db    "TRDOS (386) Kernel v2.0.4 by Erdogan Tan"
3504 00017D9E 3629204B65726E656C-
3504 00017DA7 2076322E302E342062-
3504 00017DB0 79204572646F67616E-
3504 00017DB9 2054616E
3505 00017DBD 00                        db    0
3506
3507                                    ; 20/02/2017
3508                                    KERNELFSIZE  equ $  ; 04/07/2016
3509
3510                                    bss_start:
3511
3512                                    ABSOLUTE bss_start
3513
3514 00017DBE ????                      alignb 8 ; 25/12/2016
3515
3516                                            ; 15/04/2016
3517                                            ; TRDOS 386 (TRDOS v2.0)
3518                                            ;     80 interrupts
3519                                            ; 11/03/2015
3520                                            ; Interrupt Descriptor Table (20/08/2014)
3521                                    idt:
3522                                            ;resb  64*8 ; INT 0 to INT 3Fh
3523                                            ; 15/04/2016
3524 00017DC0 <res 280h>                       resb   80*8 ; INT 0 to INT 4Fh
3525
3526                                    idt_end:
3527
3528                                    ;alignb 4
3529
3530                                    task_state_segment:
3531                                            ; 24/03/2015
3532 00018040 ????                      tss.link:   resw 1
3533 00018042 ????                                  resw 1
3534                                    ; tss offset 4
3535 00018044 ????????                  tss.esp0:   resd 1
3536 00018048 ????                      tss.ss0:    resw 1
3537 0001804A ????                                  resw 1
3538 0001804C ????????                  tss.esp1:   resd 1
3539 00018050 ????                      tss.ss1:    resw 1
3540 00018052 ????                                  resw 1
3541 00018054 ????????                  tss.esp2:   resd 1
3542 00018058 ????                      tss.ss2:    resw 1
3543 0001805A ????                                  resw 1
3544                                    ; tss offset 28
3545 0001805C ????????                  tss.CR3:    resd 1
3546 00018060 ????????                  tss.eip:    resd 1
3547 00018064 ????????                  tss.eflags: resd 1
3548                                    ; tss offset 40
3549 00018068 ????????                  tss.eax:    resd 1
3550 0001806C ????????                  tss.ecx:    resd 1
3551 00018070 ????????                  tss.edx:    resd 1
```

```
3552 00018074 ????????                tss.ebx:    resd 1
3553 00018078 ????????                tss.esp:    resd 1
3554 0001807C ????????                tss.ebp:    resd 1
3555 00018080 ????????                tss.esi:    resd 1
3556 00018084 ????????                tss.edi:    resd 1
3557                                  ; tss offset 72
3558 00018088 ????                    tss.ES:     resw 1
3559 0001808A ????                                resw 1
3560 0001808C ????                    tss.CS:     resw 1
3561 0001808E ????                                resw 1
3562 00018090 ????                    tss.SS:     resw 1
3563 00018092 ????                                resw 1
3564 00018094 ????                    tss.DS:     resw 1
3565 00018096 ????                                resw 1
3566 00018098 ????                    tss.FS:     resw 1
3567 0001809A ????                                resw 1
3568 0001809C ????                    tss.GS:     resw 1
3569 0001809E ????                                resw 1
3570 000180A0 ????                    tss.LDTR:   resw 1
3571 000180A2 ????                                resw 1
3572                                  ; tss offset 100
3573 000180A4 ????                                resw 1
3574 000180A6 ????                    tss.IOPB:   resw 1
3575                                  ; tss offset 104
3576                                  tss_end:
3577
3578 000180A8 ????????                k_page_dir:  resd 1 ; Kernel's (System) Page Directory address
3579                                                   ; (Physical address = Virtual address)
3580 000180AC ????????                memory_size: resd 1 ; memory size in pages
3581 000180B0 ????????                free_pages:  resd 1 ; number of free pages
3582 000180B4 ????????                next_page:   resd 1 ; offset value in M.A.T. for
3583                                                   ; first free page search
3584 000180B8 ????????                last_page:   resd 1 ; offset value in M.A.T. which
3585                                                   ; next free page search will be
3586                                                   ; stopped after it. (end of M.A.T.)
3587 000180BC ????????                first_page:  resd 1 ; offset value in M.A.T. which
3588                                                   ; first free page search
3589                                                   ; will be started on it. (for user)
3590 000180C0 ????????                mat_size:    resd 1 ; Memory Allocation Table size in pages
3591
3592                                  ; 20/11/2020
3593                                  ;vbe2bios:    resw 1 ; VBE2 video bios ID (bochs/qemu)
3594                                  ;                    ; (0B0C4h or 0B0C5h for bochs/plex86 vgabios)
3595
3596                                  ; 02/09/2014 (Retro UNIX 386 v1)
3597                                  ; 04/12/2013 (Retro UNIX 8086 v1)
3598 000180C4 ????                    CRT_START:   resw 1         ; starting address in regen buffer
3599                                                 ; NOTE: active page only
3600 000180C6 <res 10h>               CURSOR_POSN: resw 8 ; cursor positions for video pages
3601                                  ACTIVE_PAGE:
3602 000180D6 ??                      ptty:       resb 1 ; current tty
3603                                  ; 01/07/2015 - 29/01/2016
3604 000180D7 ??                      ccolor:     resb 1 ; current color attribute
3605                                  ; 26/10/2015
3606                                  ; 07/09/2014
3607 000180D8 <res 14h>               ttychr:      resw ntty+2 ; Character buffer (multiscreen)
3608
3609                                  ; 18/05/2015 (03/06/2013 - Retro UNIX 8086 v1 feature only!)
3610 000180EC ????????                p_time:      resd 1     ; present time (for systime & sysmdate)
3611
3612                                  ; 18/05/2015 (16/08/2013 - Retro UNIX 8086 v1 feature only !)
3613                                  ; (open mode locks for pseudo TTYs)
3614                                  ; [ major tty locks (return error in any conflicts) ]
3615 000180F0 <res 14h>               ttyl:        resw ntty+2 ; opening locks for TTYs.
3616
3617                                  ; 15/04/2015 (Retro UNIX 386 v1)
3618                                  ; 22/09/2013 (Retro UNIX 8086 v1)
3619 00018104 <res Ah>                wlist:       resb ntty+2 ; wait channel list (0 to 9 for TTYs)
3620                                  ; 15/04/2015 (Retro UNIX 386 v1)
3621                                  ;; 12/07/2014 -> sp_init set comm. parameters as 0E3h
3622                                  ;; 0 means serial port is not available
3623                                  ;;comprm: ; 25/06/2014
3624 0001810E ??                      com1p:       resb 1  ;;0E3h
3625 0001810F ??                      com2p:       resb 1  ;;0E3h
3626
3627                                  ; 17/11/2015
3628                                  ; request for response (from the terminal)
3629 00018110 ????                    req_resp:    resw 1
3630                                  ; 07/11/2015
3631 00018112 ??                      ccomport:    resb 1 ; current COM (serial) port
3632                                                   ; (0= COM1, 1= COM2)
3633                                  ; 09/11/2015
3634 00018113 ??                      comqr:       resb 1 ; 'query or response' sign (u9.s, 'sndc')
3635                                  ; 07/11/2015
3636 00018114 ????                    rchar:       resw 1 ; last received char for COM 1 and COM 2
3637 00018116 ????                    schar:       resw 1 ; last sent char for COM 1 and COM 2
3638
3639                                  ; 22/08/2014 (RTC)
3640                                  ; (Packed BCD)
3641 00018118 ??                      time_seconds: resb 1
3642 00018119 ??                      time_minutes: resb 1
3643 0001811A ??                      time_hours:   resb 1
3644 0001811B ??                      date_wday:    resb 1
3645 0001811C ??                      date_day:     resb 1
3646 0001811D ??                      date_month:   resb 1
3647 0001811E ??                      date_year:    resb 1
3648 0001811F ??                      date_century: resb 1
3649
3650                                  ; 24/01/2016
3651 00018120 ????????                RTC_LH:         resd 1
3652 00018124 ??                      RTC_WAIT_FLAG: resb 1
3653 00018125 ??                      USER_FLAG:     resb 1
3654                                  ; 19/05/2016
3655                                  ;RTC_second:
3656 00018126 ??                      RTC_2Hz:       resb 1 ;  from 2Hz interrupt to 1Hz timer event function
```

```
                                    %include 'diskbss.s'     ; UNINITIALIZED DISK (BIOS) DATA
                                <1> ; ********************************************************************************
                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - diskbss.s
                                <1> ; --------------------------------------------------------------------------------
                                <1> ; Last Update: 24/01/2016 (11/04/2021)
                                <1> ; --------------------------------------------------------------------------------
                                <1> ; Beginning: 24/01/2016
                                <1> ; --------------------------------------------------------------------------------
                                <1> ; Assembler: NASM version 2.11 (trdos386.s)
                                <1> ; --------------------------------------------------------------------------------
                                <1> ; Turkish Rational DOS
                                <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
                                <1> ;
                                <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
                                <1> ; diskbss.inc (10/07/2015)
                                <1> ;
                                <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
                                <1> ; ********************************************************************************
                                <1>
                                <1> ; Retro UNIX 386 v1 Kernel - DISKBSS.INC
                                <1> ; Last Modification: 10/07/2015
                                <1> ;     (Unitialized Disk Parameters Data section for 'DISKIO.INC')
                                <1>
00018127 ??                     <1> alignb 2
                                <1>
                                <1> ;----------------------------------------
                                <1> ;     TIMER DATA AREA           :
                                <1> ;----------------------------------------
                                <1>
                                <1> TIMER_LH:    ; 16/02/205
00018128 ????                   <1> TIMER_LOW:      resw     1              ; LOW WORD OF TIMER COUNT
0001812A ????                   <1> TIMER_HIGH:     resw     1              ; HIGH WORD OF TIMER COUNT
0001812C ??                     <1> TIMER_OFL:      resb     1              ; TIMER HAS ROLLED OVER SINCE LAST READ
                                <1>
                                <1> ;----------------------------------------
                                <1> ;     DISKETTE DATA AREAS       :
                                <1> ;----------------------------------------
                                <1>
0001812D ??                     <1> SEEK_STATUS:       resb   1
0001812E ??                     <1> MOTOR_STATUS:      resb   1
0001812F ??                     <1> MOTOR_COUNT:       resb   1
00018130 ??                     <1> DSKETTE_STATUS:    resb   1
00018131 ??????????????         <1> NEC_STATUS:        resb   7
                                <1>
                                <1> ;----------------------------------------
                                <1> ;     ADDITIONAL MEDIA DATA           :
                                <1> ;----------------------------------------
                                <1>
00018138 ??                     <1> LASTRATE:    resb   1
00018139 ??                     <1> HF_STATUS:   resb   1
0001813A ??                     <1> HF_ERROR:    resb   1
0001813B ??                     <1> HF_INT_FLAG: resb   1
0001813C ??                     <1> HF_CNTRL:    resb   1
0001813D ????????               <1> DSK_STATE:   resb   4
00018141 ????                   <1> DSK_TRK:     resb   2
                                <1>
                                <1> ;----------------------------------------
                                <1> ;     FIXED DISK DATA AREAS           :
                                <1> ;----------------------------------------
                                <1>
00018143 ??                     <1> DISK_STATUS1:      resb   1            ; FIXED DISK STATUS
00018144 ??                     <1> HF_NUM:            resb   1            ; COUNT OF FIXED DISK DRIVES
00018145 ??                     <1> CONTROL_BYTE:      resb   1            ; HEAD CONTROL BYTE
                                <1> ;@PORT_OFF         resb   1            ; RESERVED (PORT OFFSET)
                                <1> ;port1_off         resb   1            ; Hard disk controller 1 - port offset
                                <1> ;port2_off         resb   1            ; Hard disk controller 2 - port offset
                                <1>
00018146 ????                   <1> alignb 4
                                <1>
                                <1> ;HF_TBL_VEC:       resd   1            ; Primary master disk param. tbl. pointer
                                <1> ;HF1_TBL_VEC:      resd   1            ; Primary slave disk param. tbl. pointer
                                <1>
                                <1> HF_TBL_VEC: ; 22/12/2014
00018148 ????????               <1> HDPM_TBL_VEC:      resd   1            ; Primary master disk param. tbl. pointer
0001814C ????????               <1> HDPS_TBL_VEC:      resd   1            ; Primary slave disk param. tbl. pointer
00018150 ????????               <1> HDSM_TBL_VEC:      resd   1            ; Secondary master disk param. tbl. pointer
00018154 ????????               <1> HDSS_TBL_VEC:      resd   1            ; Secondary slave disk param. tbl. pointer
                                <1>
                                <1> ; 03/01/2015
00018158 ??                     <1> LBAMode:           resb   1
                                <1>
                                <1> ; ********************************************************************************
                                
                                    ;;; Real Mode Data (10/07/2015 - BSS)
                                
                                    ;alignb 2
                                
                                    ; 10/01/2016
                                    %include 'trdoskx.s'     ; UNINITIALIZED KERNEL (Logical Drive & FS) DATA
                                <1> ; ********************************************************************************
                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.4) - UNINITIALIZED DATA : trdoskx.s
                                <1> ; --------------------------------------------------------------------------------
                                <1> ; Last Update: 17/04/2021
                                <1> ; --------------------------------------------------------------------------------
                                <1> ; Beginning: 04/01/2016
                                <1> ; --------------------------------------------------------------------------------
                                <1> ; Assembler: NASM version 2.11 (trdos386.s)
                                <1> ; --------------------------------------------------------------------------------
                                <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
                                <1> ; TRDOS2.ASM (09/11/2011)
                                <1> ; ********************************************************************************
                                <1> ; DRV_INIT.ASM [26/09/2009] Last Update: 07/08/2011
                                <1> ; MAINPROG.ASM [17/01/2004] Last Update: 09/11/2011
                                <1> ; DIR.ASM      [17/01/2004] Last Update: 09/10/2011
                                <1> ; CMD_INTR.ASM [29/01/2005] Last update: 09/11/2011
```

```
3682                            <1> ; DRV_FAT.ASM  [07/07/2009] Last update: 21/08/2011
3683                            <1>
3684 00018159 ??????            <1> alignb 4
3685                            <1>
3686                            <1> ; MAINPROG.ASM
3687 0001815C ????????          <1> MainProgCfg_FileSize:   resd 1 ; 14/04/2016
3688 00018160 ????????          <1> MainProgCfg_LineOffset: resd 1 ; 14/04/2016
3689                            <1>
3690 00018164 ????????          <1> Current_VolSerial: resd 1
3691                            <1>
3692 00018168 ????????          <1> Current_Dir_FCluster: resd 1
3693                            <1>
3694 0001816C ??                <1> Current_Dir_Level: resb 1
3695 0001816D ??                <1> Current_FATType:   resb 1
3696 0001816E ??                <1> Current_Drv:       resb 1
3697 0001816F ??                <1> Current_Dir_Drv:   resb 1 ; '?'
3698 00018170 ??                <1>                    resb 1 ; ':'
3699 00018171 ??                <1> Current_Dir_Root:  resb 1 ; '/'
3700 00018172 <res 5Ah>          <1> Current_Directory: resb 90
3701 000181CC ??                <1> End_Of_Current_Dir_Str: resb 1
3702 000181CD ??                <1> Current_Dir_StrLen: resb 1
3703                            <1>
3704 000181CE ??                <1> CursorColumn:      resb 1
3705 000181CF ??                <1> CmdArgStart:       resb 1
3706                            <1>
3707                            <1> ; 03/02/2016
3708 000181D0 <res 4Eh>          <1> Remark:            resb 78
3709                            <1>
3710 0001821E <res 50h>          <1> CommandBuffer:     resb 80
3711                            <1>
3712 0001826E <res 100h>         <1> TextBuffer:  resb 256
3713                            <1>
3714                            <1> MasterBootBuff:
3715 0001836E <res 1BEh>         <1> MasterBootCode: resb 1BEh
3716 0001852C <res 40h>          <1> PartitionTable: resb 64
3717 0001856C ????               <1> MBIDCode:   resw 1
3718                            <1>
3719                            <1> PTable_Buffer:
3720 0001856E <res 40h>          <1> PTable_hd0: resb 64
3721 000185AE <res 40h>          <1> PTable_hd1: resb 64
3722 000185EE <res 40h>          <1> PTable_hd2: resb 64
3723 0001862E <res 40h>          <1> PTable_hd3: resb 64
3724                            <1> ; 15/07/2020
3725                            <1> ;PTable_ep0: resb 64
3726                            <1> ;PTable_ep1: resb 64
3727                            <1> ;PTable_ep2: resb 64
3728                            <1> ;PTable_ep3: resb 64
3729                            <1>
3730                            <1> ; 13/08/2020
3731 0001866E ??                <1> scount: resb 1 ; 16/05/2016 (diskio.s, 'int33h:')
3732 0001866F ??                <1>       resb 1
3733 00018670 ??                <1>       resb 1
3734 00018671 ??                <1>       resb 1
3735                            <1>
3736 00018672 ??                <1> HD_LBA_yes: resb 1
3737 00018673 ??                <1> PP_Counter: resb 1
3738 00018674 ??                <1> EP_Counter: resb 1
3739                            <1> ; 13/08/2020
3740 00018675 ??                <1> LD_Counter: resb 1
3741                            <1>
3742                            <1> ; 30/08/2020
3743 00018676 ????????          <1> MBR_EP_StartSector: resd 1
3744                            <1>             ; Extd partition start sector as in MBR
3745 0001867A ????????          <1> EP_StartSector: resd 1 ; next extd partition start sector
3746                            <1>       ; 15/07/2020
3747                            <1>             ;resd 1
3748                            <1>             ;resd 1
3749                            <1>
3750                            <1> ; 20/07/2020
3751 0001867E <res 200h>         <1> DOSBootSectorBuff: resb 512
3752                            <1> ; 15/07/2020
3753                            <1> ;DOSBootSectorBuff: resb 446 ; 1BEh
3754                            <1> ;MiniPartitionTable: resb 64 ;  40h
3755                            <1> ;MiniPartitionMagic: resw  1 ;  02h
3756                            <1>
3757                            <1> FAT_BuffDescriptor:
3758 0001887E ????????          <1> FAT_CurrentCluster: resd 1
3759 00018882 ??                <1> FAT_BuffValidData: resb 1
3760 00018883 ??                <1> FAT_BuffDrvName: resb 1
3761 00018884 ????              <1> FAT_BuffOffset: resw 1
3762 00018886 ????????          <1> FAT_BuffSector: resd 1
3763                            <1>
3764 0001888A ????????          <1> FAT_ClusterCounter: resd 1
3765 0001888E ????????          <1> LastCluster: resd 1
3766                            <1>
3767                            <1> ; 16/05/2016
3768                            <1> ;; 18/03/2016 (TRDOS v2.0)
3769                            <1> ;ClusterBuffer_Valid: resb 1
3770                            <1>
3771                            <1> Dir_BuffDescriptor:
3772 00018892 ??                <1> DirBuff_DRV: resb 1
3773 00018893 ??                <1> DirBuff_FATType: resb 1
3774 00018894 ??                <1> DirBuff_ValidData: resb 1
3775 00018895 ????              <1> DirBuff_CurrentEntry: resw 1
3776 00018897 ????              <1> DirBuff_LastEntry: resw 1
3777 00018899 ????????          <1> DirBuff_Cluster: resd 1
3778 0001889D ????              <1> DirBuffer_Size: resw 1
3779                            <1> ;DirBuff_EntryCounter: resw 1
3780                            <1>
3781                            <1> ; 01/02/2016
3782                            <1> ; these are on (real mode) segment 8000h and later
3783                            <1> ; FAT_Buffer:      resb 1536 ; 3 sectors
3784                            <1> ; Dir_Buffer:      resb 512*32
3785                            <1> ; Logical_DOSDisks:  resb 6656 ; 26 * 256 bytes
3786                            <1>
```

```
3787                            <1> ; 18/01/2016
3788                            <1>
3789 0001889F ????????          <1> FreeClusterCount: resd 1
3790                            <1>
3791 000188A3 ????????          <1> VolSize_Unit1:   resd 1
3792 000188A7 ????????          <1> VolSize_Unit2:   resd 1
3793                            <1>
3794 000188AB ????????          <1> Vol_Tot_Sec_Str_Start:    resd 1
3795 000188AF <res Ah>          <1> Vol_Tot_Sec_Str:          resb 10
3796 000188B9 ??                <1> Vol_Tot_Sec_Str_End:      resb 1
3797 000188BA ??                <1> resb 1
3798 000188BB ????????          <1> Vol_Free_Sectors_Str_Start: resd 1
3799 000188BF <res Ah>          <1> Vol_Free_Sectors_Str:       resb 10
3800 000188C9 ??                <1> Vol_Free_Sectors_Str_End:   resb 1
3801                            <1>
3802                            <1> ; 10/02/2016
3803 000188CA ??                <1> RUN_CDRV: resb 1 ; CMD_INTR.ASM  ; 09/11/2011
3804                            <1>
3805                            <1> ; 24/01/2016
3806 000188CB <res 80h>         <1> PATH_Array:     resb 128 ; DIR.ASM ; 09/10/2011
3807                            <1> ; 06/02/2016
3808 0001894B ????????          <1> CCD_DriveDT: resd 1 ; DIR.ASM ; (word)
3809 0001894F ??                <1> CCD_Level:   resb 1 ; DIR.ASM
3810 00018950 ??                <1> Last_Dir_Level:    resb 1 ; DIR.ASM
3811                            <1> ;
3812 00018951 ????              <1> CDLF_AttributesMask: resw 1 ; DIR.ASM
3813 00018953 ????????          <1> CDLF_FNAddress:    resd 1 ; DIR.ASM (word)
3814 00018957 ????              <1> CDLF_DEType: resw 1 ; DIR.ASM
3815                            <1> ;
3816 00018959 ??                <1> CD_COMMAND:  resb 1 ; DIR.ASM
3817                            <1>
3818 0001895A ????              <1> alignb 4
3819                            <1>
3820                            <1> ; 29/01/2016
3821 0001895C ??                <1> Program_Exit:      resb 1 ; CMD_INTR.ASM  ; 09/11/2011
3822                            <1>
3823                            <1> ;alignb 4
3824                            <1> ; 23/02/2016
3825 0001895D ??                <1> disk_rw_op:  resb 1 ;  0 = disk read, 1 = disk write
3826                            <1> ;disk_rw_spt:      resb 1 ; sectors per track (<= 63) /// (<256)
3827                            <1> ; 31/01/2016
3828 0001895E ??                <1> retry_count:       resb 1 ; DISK_IO.ASM ; 20/07/2011 (CHS_RetryCount)
3829 0001895F ??                <1> disk_rw_err:       resb 1 ; DISK_IO.ASM ; (Disk_IO_err_code)
3830 00018960 ????????          <1> sector_count:      resd 1 ; DISK_IO.ASM ; (Disk_RW_SectorCount)
3831                            <1>
3832                            <1> ; 06/02/2016 (long name)
3833 00018964 ????              <1> FDE_AttrMask:         resw 1 ; DIR.ASM
3834 00018966 ????              <1> AmbiguousFileName: resw 1 ; DIR.ASM
3835 00018968 ??                <1> PreviousAttr:         resb 1 ; DIR.ASM
3836                            <1> ;
3837 00018969 ??                <1> LongNameFound:   resb 1     ; DIR.ASM
3838 0001896A ??                <1> LFN_EntryLength: resb 1   ; DIR.ASM
3839 0001896B ??                <1> LFN_CheckSum:    resb 1   ; DIR.ASM
3840 0001896C <res 84h>         <1> LongFileName:    resb 132 ; DIR.ASM
3841                            <1>
3842                            <1> ;PATH_Array_Ptr: resw 1 ; DIR.ASM
3843 000189F0 ??                <1> PATH_CDLevel:       resb 1 ; DIR.ASM
3844 000189F1 ??                <1> PATH_Level:   resb 1 ; DIR.ASM
3845                            <1>
3846                            <1> ; 07/02/2016
3847 000189F2 <res Dh>          <1> Dir_File_Name:     resb 13 ; DIR.ASM ; 09/10/2011
3848                            <1>
3849                            <1> ; 10/02/2016
3850 000189FF <res Dh>          <1> Dir_Entry_Name:    resb 13 ; DIR.ASM
3851                            <1>
3852                            <1> alignb 2
3853                            <1>
3854 00018A0C ????              <1> AttributesMask: resw 1 ; CMD_INTR.ASM ; 09/11/2011
3855                            <1>
3856                            <1> ; 10/02/2016 (128 bytes -> 126 bytes)
3857                            <1> ; 08/02/2016
3858                            <1> ;FFF Structure (128 bytes) ; DIR.ASM ; 09/10/2011
3859 00018A0E ??                <1> FindFile_Drv:             resb 1
3860 00018A0F <res 41h>         <1> FindFile_Directory:       resb 65
3861 00018A50 <res Dh>          <1> FindFile_Name:            resb 13
3862                            <1> FindFile_LongNameEntryLength:
3863 00018A5D ??                <1> FindFile_LongNameYes:     resb 1 ; Sign for longname procedures
3864                            <1> ;Above 80 bytes form
3865                            <1> ;TR-DOS Source/Destination File FullName Format/Structure
3866 00018A5E ????              <1> FindFile_AttributesMask:  resw 1
3867 00018A60 <res 20h>         <1> FindFile_DirEntry:        resb 32
3868 00018A80 ????????          <1> FindFile_DirFirstCluster: resd 1
3869 00018A84 ????????          <1> FindFile_DirCluster:      resd 1
3870 00018A88 ????              <1> FindFile_DirEntryNumber:  resw 1
3871 00018A8A ????              <1> FindFile_MatchCounter:    resw 1
3872 00018A8C ????              <1> FindFile_Reserved:        resw 1 ; 06/03/2016
3873                            <1>
3874 00018A8E ????????          <1> First_Path_Pos: resd 1    ; DIR.ASM ; 09/10/2011
3875 00018A92 ????????          <1> Last_Slash_Pos: resd 1    ; DIR.ASM
3876                            <1>
3877                            <1> ; 10/02/2016
3878 00018A96 ????              <1> File_Count:     resw 1    ; DIR.ASM ; 09/10/2011
3879 00018A98 ????              <1> Dir_Count:      resw 1
3880 00018A9A ????????          <1> Total_FSize:    resd 1
3881 00018A9E ????????          <1> TFS_Dec_Begin:  resd 1
3882 00018AA2 <res Ah>          <1>                 resb 10
3883 00018AAC ??                <1> TFS_Dec_End:    resb 1
3884                            <1>
3885 00018AAD ??                <1> PrintDir_RowCounter: resb 1
3886                            <1>
3887 00018AAE ????              <1> alignb 4
3888                            <1> ; 15/02/2015 ('show' command variables)
3889 00018AB0 ????????          <1> Show_FDT:          resd 1
3890 00018AB4 ????????          <1> Show_LDDDT:        resd 1
3891 00018AB8 ????????          <1> Show_Cluster:      resd 1
```

```
3892 00018ABC ????????            <1> Show_FileSize:     resd 1
3893 00018AC0 ????????            <1> Show_FilePointer:  resd 1
3894 00018AC4 ????                <1> Show_ClusterPointer: resw 1
3895 00018AC6 ????                <1> Show_ClusterSize:  resw 1
3896 00018AC8 ??                  <1> Show_RowCount:     resb 1
3897                              <1>
3898 00018AC9 ??????              <1> alignb 4
3899                              <1> ; 21/02/2016
3900 00018ACC ????????            <1> DelFile_FNPointer: resd 1 ; ; CMD_INTR.ASM (word) ; 09/11/2011
3901                              <1> ; 27/02/2016
3902                              <1> ; DIR.ASM (09/10/2011)
3903 00018AD0 ????????            <1> DelFile_FCluster:  resd 1
3904 00018AD4 ????                <1> DelFile_EntryCounter:    resw 1
3905 00018AD6 ??                  <1> DelFile_LNEL:            resb 1
3906 00018AD7 ??                  <1> resb 1
3907                              <1>
3908                              <1> ; DIR.ASM
3909 00018AD8 ????????            <1> mkdir_DirName_Offset:    resd 1
3910 00018ADC ????????            <1> mkdir_FFCluster:         resd 1
3911 00018AE0 ????????            <1> mkdir_LastDirCluster:    resd 1
3912 00018AE4 ????????            <1> mkdir_FreeSectors:       resd 1
3913 00018AE8 ????                <1> mkdir_attrib:            resw 1
3914 00018AEA ??                  <1> mkdir_SecPerClust:       resb 1
3915 00018AEB ??                  <1> mkdir_add_new_cluster:   resb 1
3916 00018AEC <res Dh>            <1> mkdir_Name:              resb 13
3917 00018AF9 ????                <1> resw 1 ; 01/03/2016
3918                              <1> ; 27/02/2016
3919 00018AFB ??                  <1> RmDir_MultiClusters:     resb 1
3920 00018AFC ????????            <1> RmDir_DirEntryOffset:    resd 1 ; 01/03/2016 (word -> dword)
3921 00018B00 ????????            <1> RmDir_ParentDirCluster:  resd 1
3922 00018B04 ????????            <1> RmDir_DirLastCluster:    resd 1
3923 00018B08 ????????            <1> RmDir_PreviousCluster:   resd 1
3924                              <1> ; 22/02/2016
3925 00018B0C ??                  <1> UPDLMDT_CDirLevel:       resb 1
3926 00018B0D ????????            <1> UPDLMDT_CDirFCluster:    resd 1
3927                              <1>
3928 00018B11 ??????              <1> alignb 4
3929                              <1> ; DRV_FAT.ASM ; 21/08/2011
3930 00018B14 ????????            <1> gffc_next_free_cluster:  resd 1
3931 00018B18 ????????            <1> gffc_first_free_cluster: resd 1
3932 00018B1C ????????            <1> gffc_last_free_cluster:  resd 1
3933                              <1>
3934                              <1> ;29/04/2016
3935                              <1> Cluster_Index:     ;resd 1
3936                              <1> ; 22/02/2016
3937 00018B20 ????????            <1> ClusterValue:      resd 1
3938                              <1> ; 04/03/2016
3939 00018B24 ??                  <1> Attributes: resb 1
3940                              <1> ;;CFS_error:  resb 1 ;; 01/03/2016
3941 00018B25 ??                  <1> resb 1
3942 00018B26 ??                  <1> CFS_OPType: resb 1
3943 00018B27 ??                  <1> CFS_Drv:    resb 1
3944 00018B28 ????????            <1> CFS_CC:     resd 1
3945 00018B2C ????????            <1> CFS_FAT32FSINFOSEC: resd 1
3946 00018B30 ????????            <1> CFS_FAT32FC: resd 1
3947                              <1>
3948                              <1> ; 27/02/2016
3949                              <1> ;alignb 4
3950 00018B34 ????????            <1> glc_prevcluster: resd 1 ; DRV_FAT.ASM (21/08/2011)
3951                              <1> ; 22/10/2016
3952 00018B38 ????????            <1> glc_index:   resd 1 ;  Last Cluster Index (22/10/2016)
3953                              <1>
3954                              <1> ; DIR.ASM
3955 00018B3C ????                <1> DLN_EntryNumber: resw 1
3956 00018B3E ??                  <1> DLN_40h:     resb 1
3957                              <1> ; 28/02/2016
3958 00018B3F ??                  <1> TCC_FATErr:  resb 1 ; DRV_FAT.ASM
3959                              <1>
3960                              <1> alignb 4
3961                              <1> ; DIR.ASM (09/10/2011)
3962 00018B40 ????                <1> LCDE_EntryIndex: resw 1 ; LCDE_EntryOffset
3963 00018B42 ????                <1> LCDE_ClusterSN:  resw 1
3964 00018B44 ????????            <1> LCDE_Cluster:    resd 1
3965 00018B48 ????????            <1> LCDE_ByteOffset: resd 1
3966                              <1>
3967                              <1> ;alignb4
3968                              <1> ; 06/03/2016 (word -> dword)
3969                              <1> ; CMD_INTR.ASM (01/08/2010)
3970 00018B4C ????????            <1> SourceFilePath:      resd 1
3971 00018B50 ????????            <1> DestinationFilePath: resd 1
3972                              <1>
3973                              <1> ;alignb 4
3974                              <1> ; 06/03/2016
3975                              <1> ; FILE.ASM (09/10/2011)
3976                              <1> ;Source File Structure (same with 'Find File' Structure)
3977 00018B54 ??                  <1> SourceFile_Drv:                 resb 1
3978 00018B55 <res 41h>           <1> SourceFile_Directory:           resb 65
3979 00018B96 <res Dh>            <1> SourceFile_Name:                resb 13
3980                              <1> SourceFile_LongNameEntryLength:
3981 00018BA3 ??                  <1> SourceFile_LongNameYes:         resb 1 ; Sign for longname procedures
3982                              <1> ;Above 80 bytes
3983                              <1> ;is TR-DOS Source File FullName Format/Structure
3984 00018BA4 ????                <1> SourceFile_AttributesMask:      resw 1
3985 00018BA6 <res 20h>           <1> SourceFile_DirEntry:            resb 32
3986 00018BC6 ????????            <1> SourceFile_DirFirstCluster:     resd 1
3987 00018BCA ????????            <1> SourceFile_DirCluster:          resd 1
3988 00018BCE ????                <1> SourceFile_DirEntryNumber:      resw 1
3989 00018BD0 ????                <1> SourceFile_MatchCounter:  resw 1
3990                              <1> ; 16/03/2016
3991 00018BD2 ??                  <1> SourceFile_SecPerClust:         resb 1
3992 00018BD3 ??                  <1> SourceFile_Reserved:            resb 1
3993                              <1> ; Above is 128 bytes
3994                              <1>
3995                              <1> ;Destination File Structure (same with 'Find File' Structure)
3996 00018BD4 ??                  <1> DestinationFile_Drv:            resb 1
```

```
3997 00018BD5 <res 41h>          <1> DestinationFile_Directory:      resb 65
3998 00018C16 <res Dh>           <1> DestinationFile_Name:           resb 13
3999                             <1> DestinationFile_LongNameEntryLength:
4000 00018C23 ??                 <1> DestinationFile_LongNameYes:    resb 1 ; Sign for longname procedures
4001                             <1> ;Above 80 bytes
4002                             <1> ;is TR-DOS Destination File FullName Format/Structure
4003 00018C24 ????               <1> DestinationFile_AttributesMask: resw 1
4004 00018C26 <res 20h>          <1> DestinationFile_DirEntry: resb 32
4005 00018C46 ????????           <1> DestinationFile_DirFirstCluster: resd 1
4006 00018C4A ????????           <1> DestinationFile_DirCluster:     resd 1
4007 00018C4E ????               <1> DestinationFile_DirEntryNumber: resw 1
4008 00018C50 ????               <1> DestinationFile_MatchCounter:   resw 1
4009                             <1> ; 16/03/2016
4010 00018C52 ??                 <1> DestinationFile_SecPerClust:    resb 1
4011 00018C53 ??                 <1> DestinationFile_Reserved: resb 1
4012                             <1> ; Above is 128 bytes
4013                             <1>
4014                             <1> ; 24/04/2016
4015 00018C54 ????               <1> resw 1
4016                             <1>
4017                             <1> ; 10/03/2016
4018                             <1> ; FILE.ASM
4019 00018C56 ??                 <1> move_cmd_phase:    resb 1
4020 00018C57 ??                 <1> msftdf_sf_df_drv:  resb 1
4021 00018C58 ????????           <1> msftdf_drv_offset: resd 1
4022                             <1>
4023                             <1> ; 11/03/2016
4024                             <1> ; DRV_FAT.ASM (21/08/2011)
4025 00018C5C ????????           <1> FAT_anc_LCluster:  resd 1
4026 00018C60 ????????           <1> FAT_anc_FFCluster: resd 1
4027                             <1>
4028                             <1> ;alignb 4
4029                             <1>
4030                             <1> ; 14/03/2016
4031                             <1> ; TRDOS 386 = TRDOS v2.0 feature only !
4032                             <1> ; 'allocate_memory_block' in 'memory.s'
4033 00018C64 ????????           <1> mem_ipg_count:     resd 1 ; page count (for contiguous allocation)
4034 00018C68 ????????           <1> mem_pg_count:      resd 1 ; page count (for count down)
4035 00018C6C ????????           <1> mem_aperture:      resd 1 ; contiguous free pages (current)
4036 00018C70 ????????           <1> mem_max_aperture: resd 1 ; maximum value of contiguous free pages
4037 00018C74 ????????           <1> mem_pg_pos:  resd 1 ; mem. position (page #) of current aperture
4038 00018C78 ????????           <1> mem_max_pg_pos: resd 1 ; mem. position (page #) of max. aperture
4039                             <1>
4040                             <1> ; 15/03/2016
4041                             <1> ; FILE.ASM ('copy_source_file_to_destination_file')
4042 00018C7C ??                 <1> copy_cmd_phase:       resb 1
4043 00018C7D ??                 <1> csftdf_rw_err:        resb 1
4044 00018C7E ??                 <1> DestinationFileFound: resb 1
4045 00018C7F ??                 <1> csftdf_cdrv:          resb 1
4046 00018C80 ????????           <1> csftdf_filesize:      resd 1
4047                             <1> ; TRDOS386 (TRDOS v2.0)
4048 00018C84 ????????           <1> csftdf_sf_mem_addr:   resd 1
4049 00018C88 ????????           <1> csftdf_sf_mem_bsize:  resd 1
4050                             <1> ;
4051                             <1>
4052 00018C8C ????????           <1> csftdf_sf_cluster:    resd 1 ; 16/03/2016
4053 00018C90 ????????           <1> csftdf_df_cluster:    resd 1
4054                             <1> ; 16/03/2016
4055 00018C94 ????????           <1> csftdf_r_size:        resd 1
4056 00018C98 ????????           <1> csftdf_w_size:        resd 1
4057 00018C9C ????????           <1> csftdf_sf_rbytes:     resd 1
4058 00018CA0 ????????           <1> csftdf_df_wbytes:     resd 1
4059 00018CA4 ??                 <1> csftdf_percentage:    resb 1
4060                             <1> ; 17/03/2016
4061 00018CA5 ??                 <1> csftdf_videopage:     resb 1
4062 00018CA6 ????               <1> csftdf_cursorpos:     resw 1
4063 00018CA8 ????????           <1> csftdf_sf_drv_dt:     resd 1
4064 00018CAC ????????           <1> csftdf_df_drv_dt:     resd 1
4065                             <1>
4066                             <1> ; 21/03/2016
4067                             <1> ; 20/03/2016
4068                             <1> ; FILE.ASM
4069 00018CB0 ????????           <1> createfile_Name_Offset:   resd 1
4070 00018CB4 ????????           <1> createfile_FreeSectors:   resd 1
4071 00018CB8 ????????           <1> createfile_size:          resd 1
4072 00018CBC ????????           <1> createfile_FFCluster:     resd 1 ; 11/03/2016
4073 00018CC0 ????????           <1> createfile_LastDirCluster: resd 1
4074 00018CC4 ????????           <1> createfile_Cluster:       resd 1
4075 00018CC8 ????????           <1> createfile_PCluster:      resd 1
4076 00018CCC ??                 <1> createfile_attrib:        resb 1
4077 00018CCD ??                 <1> createfile_SecPerClust:   resb 1
4078 00018CCE ????               <1> createfile_DirIndex:      resw 1
4079 00018CD0 ????????           <1> createfile_CCount:        resd 1
4080 00018CD4 ????               <1> createfile_BytesPerSec:   resw 1 ; 23/03/2016
4081 00018CD6 ??                 <1> createfile_wfc:           resb 1
4082 00018CD7 ??                 <1> createfile_UpdatePDir:    resb 1 ; 31/03/2016
4083                             <1>
4084                             <1> ;alignb 4
4085                             <1>
4086                             <1> ; 11/04/2016
4087 00018CD8 ????               <1> env_var_length:       resw 1
4088                             <1>
4089 00018CDA ????               <1> alignb 4
4090                             <1>
4091                             <1> ; 25/04/2016
4092 00018CDC ??                 <1> readi.valid:       resb 1 ; valid data (>0 = valid for readi)
4093 00018CDD ??                 <1> readi.drv:         resb 1 ; drive number (0, 1,2,3,4..)
4094 00018CDE ??                 <1> readi.spc:         resb 1 ; sectors per cluster for 'readi' drive
4095 00018CDF ??                 <1> readi.s_index:     resb 1 ; sector index in current cluster (buffer)
4096 00018CE0 ????????           <1> readi.sector:      resd 1 ; current disk sector
4097 00018CE4 ????               <1> readi.bpc:         resw 1 ; bytes per cluster - 1
4098 00018CE6 ????               <1> readi.offset:      resw 1 ; byte offset in cluster buffer
4099 00018CE8 ????????           <1> readi.cluster:     resd 1 ; current cluster number
4100 00018CEC ????????           <1> readi.c_index:     resd 1 ; cluster index of the current cluster (0,1,2,3..)
4101 00018CF0 ????????           <1> readi.fclust:      resd 1 ; first cluster of the current cluster
```

```
4102 00018CF4 ????????            <1> readi.fs_index:    resd 1 ; sector index in disk/file section (for Singlix FS)
4103                              <1> ;readi.buffer:     resd 1 ; readi sector buffer address
4104                              <1>
4105                              <1> ;alignb 4
4106                              <1>
4107 00018CF8 ??                  <1> writei.valid:      resb 1 ; valid data (>0 = valid for writei)
4108 00018CF9 ??                  <1> writei.drv:        resb 1 ; drive number (0, 1,2,3,4..)
4109 00018CFA ??                  <1> writei.spc:        resb 1 ; sectors per cluster for 'writei' drive
4110 00018CFB ??                  <1> writei.s_index:    resb 1 ; sector index in current cluster (buffer)
4111 00018CFC ????????            <1> writei.sector:     resd 1 ; current disk sector
4112 00018D00 ????                <1> writei.bpc:        resw 1 ; bytes per cluster - 1
4113 00018D02 ????                <1> writei.offset:     resw 1 ; byte offset in cluster buffer
4114 00018D04 ????????            <1> writei.cluster:    resd 1 ; current cluster number
4115 00018D08 ????????            <1> writei.c_index:    resd 1 ; cluster index of the current cluster (0,1,2,3..)
4116 00018D0C ????????            <1> writei.fclust:     resd 1 ; first cluster of the current cluster
4117 00018D10 ????????            <1> writei.fs_index:   resd 1 ; sector index in disk/file section (for Singlix FS)
4118                              <1> ;writei.buffer:    resd 1 ; writei sector buffer address
4119 00018D14 ????????            <1> writei.lclust:     resd 1 ; writei last cluster (mget_w) ; 23/10/2016
4120 00018D18 ????????            <1> writei.l_index:    resd 1 ; writei last cluster index (mget_w) ; 23/10/2016
4121 00018D1C ??                  <1> writei.ofn:        resb 1 ; open file number (to be written) ; 23/10/2016
4122                              <1>
4123 00018D1D ??????              <1> alignb 4
4124                              <1>
4125                              <1> ; 29/04/2016
4126 00018D20 ????????            <1> Run_CDirFC:        resd 1
4127 00018D24 ??                  <1> Run_Auto_Path:     resb 1
4128 00018D25 ??                  <1> Run_Manual_Path:   resb 1 ; 0 -> auto path sequence needed
4129 00018D26 ??                  <1> EXE_ID:            resb 1
4130 00018D27 ??                  <1> EXE_dot:           resb 1
4131                              <1>
4132                              <1> ; 06/05/2016
4133 00018D28 ????????            <1> mainprog_return_addr: resd 1
4134 00018D2C ????????            <1> last_error:  resd 1  ; this will be used to return error code to MainProg
4135                              <1>                      ; 'lasterror' keyword will be used later to get the
4136                              <1>                      ; last error code/number/status.
4137                              <1> ; 12/05/2016
4138 00018D30 ????????            <1> video_eax:   resd 1  ; eax return value of video function
4139                              <1>
4140                              <1> ; 01/06/2016
4141 00018D34 ????????            <1> user_buffer: resd 1  ; 'diskio.s' (INT 33h, Function 08h, floppy disk type)
4142                              <1>
4143                              <1> ; 21/05/2016 - TRDOS 386 ('swap/switch', 'rswap', [u.pri])
4144 00018D38 ??                  <1> priority:    resb 1  ; running priority level of process (0,1,2)
4145                              <1>                      ; (run queue which is process comes from)
4146                              <1> ; 22/05/2016 - TRDOS 386 ('set_run_sequence', 'rtc_int', 'u_timer')
4147 00018D39 ??                  <1> p_change:    resb 1  ; process change status (for timer events)
4148                              <1> ; 23/05/2016 - TRDOS 386 ('clock')
4149 00018D3A ??                  <1> multi_tasking: resb 1 ; Multi Tasking status (0 = disabled, >0 = enabled)
4150                              <1>                      ; (EBX will return with user buffer addr or disk type)
4151                              <1> ; 07/06/2016
4152 00018D3B ??                  <1> timer_events:      resb 1  ; number of (active) timer events, <= 16
4153                              <1>
4154                              <1> ; 24/06/2016
4155 00018D3C ??                  <1> w_str_cmd:   resb 1 ; WRITE_STRING command (0,1,2,3) ; video.s
4156 00018D3D ??                  <1> p_crt_mode:  resb 1  ; previous video mode (=3 or 0), backup mark/sign
4157                              <1> ; 26/06/2016
4158 00018D3E ??                  <1> p_crt_page:  resb 1  ; previous active page (for 'set_mode')
4159                              <1> ; 04/07/2016
4160 00018D3F ??                  <1> noclearmem:  resb 1  ; if set, 'SET MODE' (INT 31h) function (AH = 4)
4161                              <1>                      ; will not clear the video memory
4162                              <1>                      ; (usable for graphics modes only)
4163                              <1> alignb 2
4164 00018D40 ????                <1> CRT_LEN:     resw 1  ; length of regen buffer in bytes
4165 00018D42 <res 10h>           <1> cursor_pposn:      resw 8  ; cursor positions backup
4166                              <1>
4167                              <1> ; 10/07/2016 ('VGA_FONT_SETUP', INT 43H address for x86 real mode bios)
4168 00018D52 ????????            <1> VGA_INT43H:  resd 1 ; 0 = default (not configured by user)
4169                              <1>                      ; 0FFFFFFFFh = user defined fonts
4170                              <1>                      ; address:
4171                              <1>                      ;      vgafont8
4172                              <1>                      ;      vgafont16
4173                              <1>                      ;      vgafont14
4174                              <1>
4175                              <1> ; 25/07/2016
4176 00018D56 ??                  <1> VGA_MTYPE:   resb 1  ; 0=CTEXT,1=MTEXT,2=CGA,3=PLANAR1,4=PLANAR4,5=LINEAR
4177                              <1>
4178                              <1> ; 23/10/2016
4179 00018D57 ??                  <1> setfmod            resb 1 ; update last modification date&time sign (if >0)
4180                              <1>                      ; (it is Open File Number + 1, if > 0)
4181                              <1> alignb 4
4182                              <1>
4183                              <1> ; 16/10/2016
4184 00018D58 ????????            <1> FFF_UBuffer: resd 1  ; User's buffer address for FFF & FNF system calls
4185                              <1> ; 15/10/2016
4186 00018D5C ??                  <1> FFF_Valid:   resb 1  ; Find First File Structure validation byte
4187                              <1>                      ; 0  = invalid (Find Next File can't use FFF struct)
4188                              <1>                      ; >0 = valid, return type for FFF and Find Next File
4189                              <1>                      ; 24 = basic parameters, 24 bytes
4190                              <1>                      ; 128 = entire FFF structure/table, 128 bytes
4191                              <1> ; 16/10/2016 (FFF_Attrib: resw 1)
4192 00018D5D ??                  <1> FFF_Attrib:  resb 1 ; Find First File attributes for Find Next File (LB)
4193 00018D5E ??                  <1> FFF_RType:   resb 1  ; FFF return type (0 = Basic, >0 = complete) (HB)
4194                              <1> ; 16/10/2016 - 05/10/2016 (Set Working Path)
4195 00018D5F ??                  <1> SWP_inv_fname:     resb 1 ; Set Working Path - Invalid File Name
4196 00018D60 ????                <1> SWP_Mode:    resw 1 ; Set Working Path - Mode
4197 00018D62 ??                  <1> SWP_DRV:     resb 1 ; Set Working Path - Drive
4198 00018D63 ??                  <1> SWP_DRV_chg: resb 1 ; Set Working Path - Drive Change
4199                              <1>
4200                              <1> ; 27/02/2017
4201 00018D64 ??                  <1> fpready:     resb 1 ; '80387 fpu is ready' flag
4202                              <1>
4203                              <1> ; 17/04/2021
4204                              <1> ; (DEVICE parameters is disabled as temporary)
4205                              <1>
4206                              <1> ; 08/10/2016
```

```
4207                            <1> ;device_name:      resb 9  ; capitalized (and zero padded) device name
4208                            <1>                    ; (example: "TTY0",0,0,0,0,0")
4209                            <1>
4210 00018D65 ??????           <1> alignb 4
4211                            <1>
4212                            <1> ; 08/10/2016
4213                            <1> ; 07/10/2016
4214                            <1> ; Table of kernel devices (which do not use installable device drivers)
4215                            <1> ; has been coded into KERNEL (trdosk9.s)
4216                            <1> ; 07/10/2016
4217                            <1> ; 8 installable device drivers available to install (NUMIDEV)
4218                            <1> ;IDEV_PGDIR: resd NUMIDEV
4219                            <1>                    ; Page directories of installable device drivers
4220                            <1>                    ;
4221                            <1>                    ; Note: Virtual start address is always 400000h
4222                            <1>                    ; (end of the 1st 4MB). [org 400000h]
4223                            <1>                    ; Segments: KCODE, KDATA
4224                            <1>                    ; Method: call 400000h (after changing page dir)
4225                            <1>                    ; Query code located at the start (400000h).
4226                            <1>                    ; Query code returns with
4227                            <1>                    ;   eax = device type and driver version
4228                            <1>                    ;         AL = Device Type minor
4229                            <1>                    ;         AH = Device Type major
4230                            <1>                    ;         Byte 16-23 : Version minor
4231                            <1>                    ;       Byte 24-31 : Version major - 1
4232                            <1>                    ;                    (0:0 -> 1.0)
4233                            <1>                    ;   ebx = initialization code address
4234                            <1>                    ;   ecx = configuration table address
4235                            <1>                    ;   edx = description table address
4236                            <1>                    ;   esi = device (default) name address (ASCIIZ)
4237                            <1>                    ;       (name has "/DEV/" prefix)
4238                            <1>                    ;   edi = dispatch table address
4239                            <1>                    ;        (for calling kernel-device functions)
4240                            <1>                    ;   ebp = address table address
4241                            <1>                    ; Initialization code returns with
4242                            <1>                    ;   eax = open code address
4243                            <1>                    ;   ecx = close code address
4244                            <1>                    ;   ebx = read code address
4245                            <1>                    ;   edx = write code address
4246                            <1>                    ;   esi = IOCTL code address
4247                            <1>                    ;   edi = dispatch table address
4248                            <1>                    ;   ebp = address table address
4249                            <1>                    ; Address Table:
4250                            <1>                    ;    Offset 0  : open code address
4251                            <1>                    ;    Offset 4  : read code address
4252                            <1>                    ;    Offset 8  : write code address
4253                            <1>                    ;    Offset 12 : close code address
4254                            <1>                    ;    Offset 16 : IOCTL code address
4255                            <1>                    ;    Offset 20 : initialization code address
4256                            <1>                    ;    Offset 24 : description table address
4257                            <1>                    ;    Offset 28 : configuration table address
4258                            <1>                    ;    Offset 32 : device name address
4259                            <1>                    ;    Offset 36 : dispatch table address
4260                            <1>                    ;          (for calling kernel-device functions)
4261                            <1>
4262                            <1> ;IDEV_NAME:  resb 8*NUMIDEV
4263                            <1>                      ; 8 byte names of installable device drivers
4264                            <1>
4265                            <1> ;IDEV_TYPE:  resb NUMIDEV ; Driver type of installable device drivers
4266                            <1> ;IDEV_FLAGS: resb NUMIDEV ; Device access parameters for installable
4267                            <1>                           ; device drivers (These values are set while
4268                            <1>                      ; the device driver is being loaded.)
4269                            <1> ;IDEV_OADDR: resd NUMIDEV ; open function addr for installable dev driver
4270                            <1> ;IDEV_CADDR: resd NUMIDEV ; close function addr for installable dev driver
4271                            <1> ;IDEV_RADDR: resd NUMIDEV ; read function addr for installable dev driver
4272                            <1> ;IDEV_WADDR: resd NUMIDEV ; write function addr for installable dev driver
4273                            <1>
4274                            <1> ; 08/10/2016
4275                            <1> ; 07/10/2016
4276                            <1> ; Device Open and Access parameters
4277                            <1> ;DEV_ACCESS: resb NUMOFDEVICES    ; bit 0 = accessable by normal users
4278                            <1>                                   ; bit 1 = read access permission
4279                            <1>                                   ; bit 2 = write access permission
4280                            <1>                                   ; bit 3 = IOCTL permission to users
4281                            <1>                                   ; bit 4 = block device if it is set
4282                            <1>                                   ; bit 5 = 16 bit or 1024 byte data
4283                            <1>                                   ; bit 6 = 32 bit or 2048 byte data
4284                            <1>                                   ; bit 7 = installable device driver
4285                            <1> ;DEV_R_OWNER:      resb NUMOFDEVICES    ; Reading owner no (u.uid) of devices
4286                            <1> ;DEV_R_OPENCOUNT: resb NUMOFDEVICES  ; Reading open count
4287                            <1> ;DEV_W_OWNER:      resb NUMOFDEVICES    ; Writing owner no (u.uid) of devices
4288                            <1> ;DEV_W_OPENCOUNT: resb NUMOFDEVICES  ; Writing open count
4289                            <1> ;DEV_DRIVER: resb NUMOFDEVICES    ; device driver number (1 to 7Fh)
4290                            <1>                                   ; *if bit 7 is set (80 to FFh)
4291                            <1>                                   ; *if it is installable device driver
4292                            <1>                                   ; *index (0 to 7Fh)
4293                            <1>                                   ; otherwise it is kernel device index
4294                            <1> ;DEV_OPENMODE:     resb NUMOFDEVICES    ; 1 = read mode
4295                            <1>                                   ; 2 = write mode
4296                            <1>                                   ; 3 = read & write
4297                            <1>                                   ; 0 = not open (free)
4298                            <1> ;DEV_NAME_PTR:     resd NUMOFDEVICES    ; pointers to name addresses of drivers
4299                            <1>                                   ; Address base: KDEV_NAME+
4300                            <1>                                   ; or IDEV_NAME+
4301                            <1> ;DEV_R_POINTER:    resd NUMOFDEVICES    ; reading pointer, writing pointer
4302                            <1> ;DEV_W_POINTER:    resd NUMOFDEVICES    ; sector number if block device
4303                            <1>                                   ; character offset if char device
4304                            <1> alignb 4
4305                            <1>
4306                            <1> ; 06/10/2016
4307                            <1> ; Open File Parameters
4308 00018D68 <res 28h>        <1> OF_FCLUSTER: resd OPENFILES  ; First clusters of open files
4309 00018D90 <res Ah>         <1> OF_DRIVE:    resb OPENFILES  ; Logical DOS drive numbers of open files
4310 00018D9A <res Ah>         <1> OF_MODE:     resb OPENFILES  ; Open mode (1 = read, 2 = write, 3 = r&w)
4311 00018DA4 <res Ah>         <1> OF_STATUS:   resb OPENFILES  ; (bit 0 = read, bit 1 = write)
```

```
4312 00018DAE <res Ah>           <1> OF_OPENCOUNT: resb OPENFILES  ; Open counts of open files
4313 00018DB8 <res 28h>          <1> OF_POINTER:  resd OPENFILES   ; File seek/read/write pointer
4314 00018DE0 <res 28h>          <1> OF_SIZE:     resd OPENFILES   ; File sizes of open files (in bytes)
4315 00018E08 <res 28h>          <1> OF_DIRFCLUSTER: resd OPENFILES ; Directory First Clusters of open files
4316 00018E30 <res 28h>          <1> OF_DIRCLUSTER: resd OPENFILES ; Directory (Entry) Clusters of open files
4317 00018E58 <res 28h>          <1> OF_VOLUMEID: resd OPENFILES  ; Vol ID for removable drives of open files
4318 00018E80 <res 28h>          <1> OF_CCLUSTER: resd OPENFILES  ; Current clusters of open files
4319 00018EA8 <res 28h>          <1> OF_CCINDEX:  resd OPENFILES  ; Cluster index numbers of current clusters
4320                             <1> ; 24/10/2016
4321 00018ED0 <res 14h>          <1> OF_DIRENTRY: resw OPENFILES  ; Directory entry index no. in dir cluster
4322                             <1>                              ; Sector index = entry index / 16
4323                             <1> ;alignb 2
4324                             <1>
4325 00018EE4 <res 60h>          <1> DTA:         resd 24         ; Find First File data transfer area
4326                             <1>
4327                             <1> ; 19/12/2016
4328 00018F44 ??                 <1> tcallback:   resb 1          ; Timer callback method flag for 'systimer'
4329 00018F45 ??                 <1> trtc:        resb 1          ; Timer interrupt type flag for 'systimer'
4330                             <1> ; 20/02/2017
4331 00018F46 ??                 <1> no_page_swap: resb 1         ; Swap lock for Signal Response Byte pages
4332                             <1> ;;15/01/2017
4333                             <1> ; 02/01/2017
4334                             <1> ;;intflg:    resb 1          ; software interrupt in progress signal
4335                             <1>                              ; (for timer interrupt)
4336 00018F47 ??                 <1> alignb 4
4337                             <1> ; 13/04/2017
4338                             <1> ;DEV_INTR:   resb NUMOFDEVICES ; Device Interrupt (IRQ) number + 1
4339                             <1>                              ; (0= not available, 1= IRQ 0, 16= IRQ 15)
4340 00018F48 <res 40h>          <1> DEV_INT_HNDLR: resd 16       ; Device Interrupt Handler addr, if > 0
4341                             <1>
4342                             <1> ;alignb 4
4343                             <1>
4344                             <1> ; 26/02/2017 ; IRQ Callback parameters ('syscalbac')
4345                             <1> ;Index: ; 0 to 8
4346                             <1> ;     0 = IRQ3, 1 = IRQ4, 2 = IRQ5, 3 = IRQ7
4347                             <1> ;     4 = IRQ9, 5 = IRQ10, 6 = IRQ11, 7 = IRQ12, 8 = IRQ13
4348 00018F88 <res 9h>           <1> IRQ.owner:   resb 9          ; owner, 0 = free, >0 = [u.uno]
4349 00018F91 <res 9h>           <1> IRQ.dev:     resb 9          ; 0 = default/kernel, >0 = device number
4350 00018F9A <res 9h>           <1> IRQ.method:  resb 9          ; 0 = Signal Response Byte, 1 = Callback
4351 00018FA3 <res 9h>           <1> IRQ.srb:     resb 9          ; Signal Response/Return Byte value
4352 00018FAC <res 24h>          <1> IRQ.addr:    resd 9          ; Rignal Response Byte address (physical)
4353                             <1>                              ; or Callback service address (virtual)
4354                             <1> ; 28/02/2017
4355 00018FD0 ????????           <1> IRQ_cr3:     resd 1          ; for saving cr3 register in IRQ handler
4356 00018FD4 ??                 <1> IRQnum:      resb 1          ; IRQ number for IRQ handler (trdosk8.s)
4357                             <1>
4358                             <1> ; 10/04/2017
4359                             <1> ; 03/04/2017
4360                             <1> ; UNINITIALIZED AUDIO DATA
4361 00018FD5 ??????            <1> alignb 4
4362 00018FD8 ??                 <1> audio_pci:   resb 1
4363 00018FD9 ??                 <1> audio_device: resb 1
4364 00018FDA ??                 <1> audio_mode:  resb 1
4365 00018FDB ??                 <1> audio_intr:  resb 1
4366 00018FDC ??                 <1> audio_busy:  resb 1          ; Busy flag for audio irq ; 21/04/2017
4367 00018FDD ??                 <1> audio_reserved: resb 1
4368 00018FDE ????               <1> audio_io_base: resw 1        ; Base I/O address of audio device
4369 00018FE0 ????????           <1> audio_dev_id: resd 1         ; BUS/DEV/FN ; 00000000BBBBBBBBDDDDDFFF00000000
4370 00018FE4 ????????           <1> audio_vendor: resd 1
4371 00018FE8 ????????           <1> audio_stats_cmd: resd 1
4372                             <1> ;
4373 00018FEC ????????           <1> audio_buffer:   resd 1       ; virtual address of user's audio buffer
4374 00018FF0 ????????           <1> audio_p_buffer: resd 1       ; Physical address of user's audio buffer
4375 00018FF4 ????????           <1> audio_buff_size: resd 1      ; user's audio buffer size (half buffer size)
4376 00018FF8 ????????           <1> audio_dma_buff: resd 1       ; dma buffer address
4377 00018FFC ????????           <1> audio_dmabuff_size: resd 1 ; dma buffer size (2 * half buffer size)
4378 00019000 ??                 <1> audio_flag:  resb 1          ; dma buffer flag (1st half = 0, 2nd half = 1)
4379 00019001 ??                 <1> audio_user:  resb 1          ; user number of the owner
4380 00019002 ??                 <1> audio_cb_mode: resb 1        ; 0 = signal response byte method
4381                             <1>                              ; 1 = callback method
4382                             <1>                              ; 2 = s.r.b. method with auto increment
4383 00019003 ??                 <1> audio_srb:   resb 1          ; signal response byte value
4384 00019004 ????????           <1> audio_cb_addr: resd 1        ; callback service address or s.r.b. address
4385                             <1>                              ; (s.r.b. addr is physical, cbs addr is virtual)
4386                             <1>
4387 00019008 ??                 <1> audio_bps:   resb 1          ; selected mode: 8 bit, 16 bit
4388 00019009 ??                 <1> audio_stmo:  resb 1          ; selected mode: mono /stereo
4389 0001900A ????               <1> audio_freq:  resw 1          ; sampling rate
4390                             <1>
4391                             <1> ; 21/04/2017
4392 0001900C ??                 <1> audio_play_cmd: resb 1       ; Play/Stop command (1 = play, 0 = stop)
4393                             <1> audio_civ: ; 28/05/2017      ; Current Buffer Index (AC'97)
4394 0001900D ??                 <1> audio_flag_eol: resb 1       ; End of Link status (vt8233, EOL/FLAG)
4395                             <1>
4396                             <1> audio_master_volume:
4397 0001900E ??                 <1> audio_master_volume_l: resb 1 ; sound volume (lineout) left channel
4398 0001900F ??                 <1> audio_master_volume_r: resb 1 ; sound volume (lineout) right channel
4399                             <1>
4400                             <1> alignb 4
4401                             <1> ; 28/05/2017
4402                             <1> ; AC'97 Audio Controller Base Adress Registers
4403 00019010 ????               <1> NAMBAR:      resw 1 ; Native Audio Mixer Base Address
4404 00019012 ????               <1> NABMBAR:     resw 1 ; Native Audio Bus Mastering Base Address
4405                             <1>
4406                             <1> ;alignb 4
4407                             <1> ; 21/04/2017
4408 00019014 <res 400h>         <1> audio_bdl_buff:    resd 32*8  ; VT8233 (AC97) BDL Buffer Size
4409                             <1> ; 12/05/2017
4410 00019414 ????????           <1> base_addr:   resd 1 ; 'direct_memory_access' (memory.s)
4411                             <1>
4412                             <1> ; 28/08/2017
4413                             <1> ; 20/08/2017
4414 00019418 ??                 <1>              resb 1  ;
4415 00019419 ??                 <1> dma_user:    resb 1 ; user number for sysdma
4416 0001941A ??                 <1> dma_channel: resb 1 ; dma channel for sysdma
```

```
4417 0001941B ??                <1> dma_mode:    resb 1  ; dma mode for sysdma
4418 0001941C ????????          <1> dma_addr:    resd 1  ; dma buffer physical addr for sysdma
4419 00019420 ????????          <1> dma_size:    resd 1  ; dma buffer size (in bytes) for sysdma
4420 00019424 ????????          <1> dma_start:   resd 1  ; dma start address for sysdma
4421 00019428 ????????          <1> dma_count:   resd 1  ; dma count (in bytes) for sysdma
4422                            <1>
4423 0001942C <res 6BD4h>       <1> alignb 65536
4424                            <1> ; 09/08/2017
4425                            <1> ; 12/05/2017
4426 00020000 <res 10000h>      <1> sb16_dma_buffer: resb 65536 ; DMA buffer for sb16 audio playing.
3666                                ; 24/01/2016
3667                                %include 'ubss.s'  ; UNINITIALIZED KERNEL (USER) DATA
3668                            <1> ; ****************************************************************************
3669                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - UNINITIALIZED USER DATA : ubss.s
3670                            <1> ; ----------------------------------------------------------------------------
3671                            <1> ; Last Update: 28/02/2017
3672                            <1> ; ----------------------------------------------------------------------------
3673                            <1> ; Beginning: 24/01/2016
3674                            <1> ; ----------------------------------------------------------------------------
3675                            <1> ; Assembler: NASM version 2.11 (trdos386.s)
3676                            <1> ; ----------------------------------------------------------------------------
3677                            <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
3678                            <1> ; ux.s (04/12/2015)
3679                            <1> ; ****************************************************************************
3680                            <1>
3681                            <1> ; Retro UNIX 386 v1 Kernel - ux.s
3682                            <1> ; Last Modification: 04/12/2015
3683                            <1> ;
3684                            <1> ; ///////// RETRO UNIX 386 V1 SYSTEM DEFINITIONS ///////////////
3685                            <1> ; (Modified from
3686                            <1> ;     Retro UNIX 8086 v1 system definitions in 'UNIX.ASM', 01/09/2014)
3687                            <1> ; ((UNIX.ASM (RETRO UNIX 8086 V1 Kernel), 11/03/2013 - 01/09/2014))
3688                            <1> ; ----------------------------------------------------------------------------
3689                            <1> ; Derived from UNIX Operating System (v1.0 for PDP-11)
3690                            <1> ; (Original) Source Code by Ken Thompson (1971-1972)
3691                            <1> ; <Bell Laboratories (17/3/1972)>
3692                            <1> ; <Preliminary Release of UNIX Implementation Document>
3693                            <1> ; (Section E10 (17/3/1972) - ux.s)
3694                            <1> ; ****************************************************************************
3695                            <1>
3696                            <1> alignb 2
3697                            <1>
3698                            <1> inode:
3699                            <1>       ; 11/03/2013.
3700                            <1>       ;Derived from UNIX v1 source code 'inode' structure (ux).
3701                            <1>       ;i.
3702                            <1>
3703 00030000 ????              <1>       i.flgs:      resw 1
3704 00030002 ??                <1>       i.nlks:      resb 1
3705 00030003 ??                <1>       i.uid:       resb 1
3706                            <1>       ;i.size:     resw 1 ; size
3707 00030004 ????              <1>                    resw 1 ; 29/04/2016
3708 00030006 <res 10h>         <1>       i.dskp:      resw 8 ; 16 bytes
3709 00030016 ????????          <1>       i.ctim:      resd 1
3710 0003001A ????????          <1>       i.mtim:      resd 1
3711 0003001E ????              <1>       i.rsvd:      resw 1 ; Reserved (ZERO/Undefined word for UNIX v1.)
3712                            <1>
3713                            <1> I_SIZE      equ $ - inode
3714                            <1>
3715                            <1> process:
3716                            <1>       ; 19/12/2016
3717                            <1>       ; 21/05/2016
3718                            <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
3719                            <1>       ; 06/05/2015 - Retro UNIX 386 v1
3720                            <1>       ; 11/03/2013 - 05/02/2014 (Retro UNIX 8086 v1)
3721                            <1>       ;Derived from UNIX v1 source code 'proc' structure (ux).
3722                            <1>       ;p.
3723                            <1>
3724 00030020 <res 20h>         <1>       p.pid:   resw nproc
3725 00030040 <res 20h>         <1>       p.ppid:  resw nproc
3726 00030060 <res 20h>         <1>       p.break: resw nproc
3727 00030080 <res 10h>         <1>       p.ttyc:  resb nproc ; console tty in Retro UNIX 8086 v1.
3728 00030090 <res 10h>         <1>       p.waitc: resb nproc ; waiting channel in Retro UNIX 8086 v1.
3729 000300A0 <res 10h>         <1>       p.link:  resb nproc
3730 000300B0 <res 10h>         <1>       p.stat:  resb nproc
3731                            <1>
3732                            <1>       ; 06/05/2015 (Retro UNIX 386 v1 feature only !)
3733 000300C0 <res 40h>         <1>       p.upage: resd nproc ; Physical address of the process's
3734                            <1>                           ; 'user' structure
3735                            <1>       ; 21/05/2016
3736                            <1>       ; 19/05/2016 (TRDOS 386 feature only!)
3737 00030100 <res 10h>         <1>       p.timer: resb nproc ; number of timer events of the processs
3738                            <1>
3739                            <1>       ; 19/12/2016
3740 00030110 <res 40h>         <1>       p.tcb:   resd nproc ; timer callback service address (if > 0)
3741                            <1>
3742                            <1> P_SIZE      equ $ - process
3743                            <1>
3744                            <1> ; fsp table (original UNIX v1)
3745                            <1> ;
3746                            <1> ;Entry
3747                            <1> ;          15                                      0
3748                            <1> ;  1     |---|--------------------------------------|
3749                            <1> ;        |r/w|       i-number of open file          |
3750                            <1> ;        |---|--------------------------------------|
3751                            <1> ;        |               device number              |
3752                            <1> ;        |------------------------------------------|
3753                            <1> ;    (*) | offset pointer, i.e., r/w pointer to file |
3754                            <1> ;        |------------------------------------------|
3755                            <1> ;        |  flag that says    | number of processes  |
3756                            <1> ;        |   file deleted     | that have file open  |
3757                            <1> ;        |------------------------------------------|
3758                            <1> ;  2     |                                          |
3759                            <1> ;        |------------------------------------------|
3760                            <1> ;        |                                          |
```

```
3761                            <1> ;        |-----------------------------------------|
3762                            <1> ;        |                                         |
3763                            <1> ;        |-----------------------------------------|
3764                            <1> ;        |                                         |
3765                            <1> ;        |-----------------------------------------|
3766                            <1> ;  3     |                                         |
3767                            <1> ;        |                                         |
3768                            <1> ;
3769                            <1> ; (*) Retro UNIX 386 v1 modification: 32 bit offset pointer
3770                            <1>
3771                            <1>
3772                            <1> ; 15/04/2015
3773 00030150 <res 1F4h>       <1> fsp:   resb nfiles * 10 ; 11/05/2015 (8 -> 10)
3774 00030344 ????             <1> idev:  resw 1 ; device number is 1 byte in Retro UNIX 8086 v1 !
3775 00030346 ????             <1> cdev:    resw 1 ; device number is 1 byte in Retro UNIX 8086 v1 !
3776                            <1> ; 18/05/2015
3777                            <1> ; 26/04/2013 device/drive parameters (Retro UNIX 8086 v1 feature only!)
3778                            <1> ; 'UNIX' device numbers (as in 'cdev' and 'u.cdrv')
3779                            <1> ;     0 -> root device (which has Retro UNIX 8086 v1 file system)
3780                            <1> ;     1 -> mounted device (which has Retro UNIX 8086 v1 file system)
3781                            <1> ; 'Retro UNIX 8086 v1' device numbers: (for disk I/O procedures)
3782                            <1> ;     0 -> fd0 (physical drive, floppy disk 1), physical drive number = 0
3783                            <1> ;     1 -> fd1 (physical drive, floppy disk 2), physical drive number = 1
3784                            <1> ;     2 -> hd0 (physical drive, hard disk 1), physical drive number = 80h
3785                            <1> ;     3 -> hd1 (physical drive, hard disk 2), physical drive number = 81h
3786                            <1> ;     4 -> hd2 (physical drive, hard disk 3), physical drive number = 82h
3787                            <1> ;     5 -> hd3 (physical drive, hard disk 4), physical drive number = 83h
3788 00030348 ??                <1> rdev:  resb 1 ; root device number ; Retro UNIX 8086 v1 feature only!
3789                            <1>               ; as above, for physical drives numbers in following table
3790 00030349 ??                <1> mdev:  resb 1 ; mounted device number ; Retro UNIX 8086 v1 feature only!
3791                            <1> ; 15/04/2015
3792 0003034A ??                <1> active:      resb 1
3793 0003034B ??                <1>              resb 1 ; 09/06/2015
3794 0003034C ????              <1> mnti:  resw 1
3795 0003034E ????              <1> mpid:  resw 1
3796 00030350 ????              <1> rootdir: resw 1
3797                            <1>
3798                            <1> ; 21/05/2016 - TRDOS 386 (TRDOS v2.0) - priority levels, 3 run queues
3799                            <1> runq:
3800 00030352 ????              <1> runq_event:   resw 1 ; high priority, 'run for event'            ; 2
3801 00030354 ????              <1> runq_normal:  resw 1 ; normal/regular priority, 'run as requlat' ; 1
3802 00030356 ????              <1> runq_background: resw 1 ; low priority, 'run on background'         ; 0
3803                            <1> ;
3804 00030358 ??                <1> imod:  resb 1
3805 00030359 ??                <1> smod:  resb 1
3806 0003035A ??                <1> mmod:  resb 1
3807 0003035B ??                <1> sysflg:      resb 1
3808                            <1>
3809                            <1> alignb 4
3810                            <1>
3811                            <1> user:
3812                            <1>       ; 13/01/2017
3813                            <1>       ; 19/12/2016
3814                            <1>       ; 21/05/2016 - TRDOS 386 (TRDOS v2.0)
3815                            <1>       ;             [u.pri] usage method modification
3816                            <1>       ; 04/12/2015
3817                            <1>       ; 18/10/2015
3818                            <1>       ; 12/10/2015
3819                            <1>       ; 21/09/2015
3820                            <1>       ; 24/07/2015
3821                            <1>       ; 16/06/2015
3822                            <1>       ; 09/06/2015
3823                            <1>       ; 11/05/2015
3824                            <1>       ; 16/04/2015 (Retro UNIX 386 v1 - 32 bit modifications)
3825                            <1>       ; 10/10/2013
3826                            <1>       ; 11/03/2013.
3827                            <1>       ;Derived from UNIX v1 source code 'user' structure (ux).
3828                            <1>       ;u.
3829                            <1>
3830 0003035C ????????          <1>       u.sp:        resd 1 ; esp (kernel stack at the beginning of 'sysent')
3831 00030360 ????????          <1>       u.usp:       resd 1 ; esp (kernel stack points to user's registers)
3832 00030364 ????????          <1>       u.r0:        resd 1 ; eax
3833 00030368 ????              <1>       u.cdir:      resw 1
3834 0003036A <res Ah>          <1>       u.fp:        resb 10
3835 00030374 ????????          <1>       u.fofp:      resd 1
3836 00030378 ????????          <1>       u.dirp:      resd 1
3837 0003037C ????????          <1>       u.namep:     resd 1
3838 00030380 ????????          <1>       u.off:       resd 1
3839 00030384 ????????          <1>       u.base:      resd 1
3840 00030388 ????????          <1>       u.count:     resd 1
3841 0003038C ????????          <1>       u.nread:     resd 1
3842 00030390 ????????          <1>       u.break:     resd 1 ; break
3843 00030394 ????              <1>       u.ttyp:      resw 1
3844                            <1>       ; 10/01/2017 (TRDOS 386, relocation and dword alignment)
3845                            <1>       ; tty number (rtty, rcvt, wtty)
3846 00030396 ??                <1>       u.ttyn:      resb 1 ; 28/07/2013 - Retro Unix 8086 v1 feature only !
3847 00030397 ??                <1>       u.resb:      resb 1 ; 10/01/2017 (TRDOS 386, temporary)
3848 00030398 <res 10h>         <1>       u.dirbuf:    resb 16 ; 04/12/2015 (10 -> 16)
3849                            <1>       ;u.pri:      resw 1 ; 14/02/2014
3850 000303A8 ??                <1>       u.quant:     resb 1 ; Retro UNIX 8086 v1 Feature only ! (uquant)
3851 000303A9 ??                <1>       u.pri:       resb 1 ; Modification: 21/05/2016 (priority levels: 0, 1, 2)
3852 000303AA ????              <1>       u.intr:      resw 1
3853 000303AC ????              <1>       u.quit:      resw 1
3854                            <1>       ;u.emt:      resw 1 ; 10/10/2013
3855                            <1>       ;u.ilgins:   resw 1 ; 10/01/2017
3856 000303AE ????              <1>       u.cdrv:      resw 1 ; cdev
3857 000303B0 ??                <1>       u.uid:       resb 1 ; uid
3858 000303B1 ??                <1>       u.ruid:      resb 1
3859 000303B2 ??                <1>       u.bsys:      resb 1
3860 000303B3 ??                <1>       u.uno:       resb 1
3861 000303B4 ????????          <1>         u.upage:   resd 1 ; 16/04/2015 - Retro Unix 386 v1 feature only !
3862 000303B8 ????????          <1>       u.pgdir:     resd 1 ; 09/03/2015 (page dir addr of process)
3863 000303BC ????????          <1>       u.ppgdir:    resd 1 ; 06/05/2015 (page dir addr of the parent process)
3864 000303C0 ????????          <1>       u.pbase:     resd 1 ; 20/05/2015 (physical base/transfer address)
3865 000303C4 ????              <1>       u.pcount:    resw 1 ; 20/05/2015 (byte -transfer- count for page)
```

```
3866                                    <1>       ;u.pncount:  resw 1
3867                                    <1>             ; 16/06/2015 (byte -transfer- count for page, 'namei', 'mkdir')
3868                                    <1>       ;u.pnbase:   resd 1
3869                                    <1>             ; 16/06/2015 (physical base/transfer address, 'namei', 'mkdir')
3870                                    <1>              ; 09/06/2015
3871 000303C6 ??                       <1>       u.kcall:     resb 1 ; The caller is 'namei' (dskr) or 'mkdir' (dskw) sign
3872 000303C7 ??                       <1>       u.brwdev:    resb 1 ; Block device number for direct I/O (bread & bwrite)
3873                                    <1>              ; 24/07/2015 - 24/06/2015
3874                                    <1>       ;u.args:     resd 1 ; arguments list (line) offset from start of [u.upage]
3875                                    <1>              ; (arg list/line is from offset [u.args] to 4096 in [u.upage])
3876                                    <1>              ; ([u.args] points to argument count -argc- address offset)
3877                                    <1>              ; 24/06/2015
3878                                    <1>       ;u.core:     resd 1 ; physical start address of user's memory space (for sys exec)
3879                                    <1>       ;u.ecore:    resd 1 ; physical end address of user's memory space (for sys exec)
3880                                    <1>       ; last error number
3881 000303C8 ????????                 <1>       u.error:     resd 1 ; 28/07/2013 - 09/03/2015
3882                                    <1>              ; Retro UNIX 8086/386 v1 feature only!
3883                                    <1>             ; 21/09/2015 (debugging - page fault analyze)
3884 000303CC ????????                 <1>       u.pfcount:   resd 1 ; page fault count for (this) process (for sys geterr)
3885                                    <1>             ; 19/12/2016 (TRDOS 386)
3886 000303D0 ????????                 <1>       u.tcb:       resd 1 ; Timer callback address/flag which will be used by timer int
3887                                    <1>             ; 13/01/2017 (TRDOS 386)
3888 000303D4 ??                       <1>       u.t_lock:    resb 1 ; Timer interrupt (callback) lock (unlocked by 'sysrele')
3889 000303D5 ??                       <1>       u.t_mode:    resb 1 ; running mode during timer interrupt (0= system, 0FFh= user)
3890                                    <1>             ; 26/02/2017 (TRDOS 386)
3891 000303D6 ??                       <1>       u.irqc:      resb 1  ; Count of IRQ callback services (IRQs in use)
3892                                    <1>             ; 28/02/2017 (TRDOS 386)
3893 000303D7 ??                       <1>       u.irqwait:   resb 1 ; IRQ waiting for callback service flag (IRQ number, If > 0)
3894 000303D8 ??                       <1>       u.r_lock:    resb 1 ; 'IRQ callback service is in progress' flag (IRQ lock)
3895 000303D9 ??                       <1>       u.r_mode:    resb 1 ; running mode during hadware interrupt
3896                                    <1>             ; 27/02/2017 (TRDOS 386)
3897 000303DA ??                       <1>       u.fpsave:    resb 1  ; TRDOS 386, 'save/restore FPU registers' flag
3898 000303DB ??                       <1> alignb 4
3899 000303DC <res 5Eh>                <1>       u.fpregs:    resb 94 ; 94 byte area for saving and restoring FPU registers
3900                                    <1>
3901 0003043A ????                     <1> alignb 4
3902                                    <1>
3903                                    <1> U_SIZE  equ $ - user
3904                                    <1>
3905                                    <1> ; 18/10/2015 - Retro UNIX 386 v1 (local variables for 'namei' and 'sysexec')
3906 0003043C ????????                 <1> pcore:  resd 1  ; physical start address of user's memory space (for sys exec)
3907 00030440 ????????                 <1> ecore:  resd 1  ; physical address of user's stack/last page (for sys exec)
3908 00030444 ????????                 <1> nbase:  resd 1  ; physical base address for 'namei' & 'sysexec'
3909 00030448 ????                     <1> ncount: resw 1  ; remain byte count in page for 'namei' & 'sysexec'
3910 0003044A ????                     <1> argc: resw 1 ; argument count for 'sysexec'
3911 0003044C ????????                 <1> argv: resd 1 ; argument list (recent) address for 'sysexec'
3912                                    <1>
3913                                    <1> ; 03/06/2015 - Retro UNIX 386 v1 Beginning
3914                                    <1> ; 07/04/2013 - 31/07/2013 - Retro UNIX 8086 v1
3915 00030450 ??                       <1> rw:    resb 1 ;; Read/Write sign (iget)
3916                                    <1>
3917                                    <1> ;alignb 4
3918                                    <1>
3919                                    <1> ; 24/04/2016
3920 00030451 ????????                 <1> ii:   resd 1 ; first cluster of the program file
3921 00030455 ????????                 <1> i.size: resd 1 ; size of the program file
3668
3669 00030459 ??????                       alignb 4
3670
3671                                        ; 23/05/2016 (TRDOS 386)
3672                                        ; 14/10/2015 (Retro UNIX 386 v1, 'unix386.s')
3673 0003045C ????????                     cr3reg:  resd 1  ; cr3 register content at the beginning of the timer
3674                                                        ; (or RTC) interrupt handler.
3675
3676                                        ; 10/12/2016 (callback)
3677                                        ; 10/06/2016
3678                                        ; 19/05/2016
3679                                        ; 18/05/2016 - TRDOS 386 feature only !
3680 00030460 <res 100h>                    timer_set: resd 16*4   ; 256 bytes memory space for 16 timer events
3681                                             ; Timer Event Structure: (max. 16 timer events, 16*16 bytes)
3682                                             ;      Owner:         resb 1 ; 0 = free
3683                                             ;                            ;>0 = process number (u.uno)
3684                                             ;      Callback:      resb 1 ; 0 = response byte address (phy)
3685                                             ;                             ; 1 = callback address (virtual)
3686                                             ;      Interrupt:     resb 1 ; 0 = Timer interrupt (or none)
3687                                             ;                            ; 1 = Real Time Clock interrupt
3688                                             ;      Response:      resb 1 ; 0 to 255, signal return value
3689                                             ;      Count Limit:   resd 1 ; count of ticks (total/set)
3690                                             ;      Current Count: resd 1 ; count of ticks (current)
3691                                             ;      Response Addr: resd 1 ; response byte (pointer) address
3692                                             ;                          ; (or callback -user service- address)
3693
3694
3695                                        ; 17/04/2021
3696                                        ; (memory page swap parameters are disabled as temporary)
3697                                        ;
3698                                        ; Memory (swap) Data (11/03/2015)
3699                                        ; 09/03/2015
3700                                        ;swpq_count: resw 1 ; count of pages on the swap queue
3701                                        ;swp_drv:    resd 1 ; logical drive description table address of the swap drive/disk
3702                                        ;swpd_size:  resd 1 ; size of swap drive/disk (volume) in sectors (512 bytes).

3703                                        ;swpd_free:  resd 1 ; free page blocks (4096 bytes) on swap disk/drive (logical)
3704                                        ;swpd_next:  resd 1 ; next free page block
3705                                        ;swpd_last:  resd 1 ; last swap page block
3706
3707                                        alignb 4
3708
3709                                        ; 10/07/2015
3710                                        ; 28/08/2014
3711 00030560 ????????                     error_code: resd 1
3712                                        ; 29/08/2014
3713 00030564 ????????                     FaultOffset: resd 1
3714                                        ; 21/09/2015
3715 00030568 ????????                     PF_Count:   resd 1 ; total page fault count
```

```
3716                                            ; (for debugging - page fault analyze)
3717                                            ; 'page_fault_handler' (memory.inc)
3718                                            ; 'sysgeterr' (u9.s)
3719
3720                                    ; 29/04/2016 (TRDOS 386 = TRDOS v2.0)
3721                                    ; 22/08/2015 (Retro UNIX 386 v1)
3722                                    buffer:
3723 0003056C ????????????????                 resb   8
3724                                    readi_buffer:
3725 00030574 <res 200h>                       resb   512
3726 00030774 ????????????????                resb   8
3727                                    writei_buffer:
3728 0003077C <res 200h>                       resb   512
3729                                    ; 24/10/2016
3730 0003097C ????????????????                resb   8
3731                                    rw_buffer:
3732 00030984 <res 800h>                       resb   2048  ; general purposed, r/w sector buffer
3733
3734                                    %if 1
3735                                    ; 17/01/2021
3736 00031184 <res 80h>                 edid_info:  resb 128 ; VESA EDID (monitor capabilities) info
3737                                    ; 28/11/2020
3738 00031204 ??                        pmi32:      resb 1 ; (>0) use VESA VBE3 protected mode calls
3739 00031205 ??                        vbe_mode_x: resb 1  ; VESA VBE3 video bios mode set options
3740 00031206 ????                      video_mode: resw 1 ; VESA VBE3 video mode (with option flags)
3741                                    ; 30/11/2020
3742 00031208 ????????                  vbe3bios_addr: resd 1 ; new (writable mem) address of VBE3 bios
3743                                    ; 02/12/2020
3744 0003120C ????????                  pmid_addr:  resd 1 ; PMInfoBlock ('PMID') linear address
3745                                    ; 14/01/2021
3746                                    ; 06/12/2020       ; VESA VBE 3 video state
3747                                    ;vbe3stbufsize: resw 1    ; video regs/dac/bios state buffer size
3748                                    ;                  ; block size in bytes
3749                                    ; 16/01/2021
3750 00031210 ????                      vbe3stbsflags: resw 1     ; video regs/dac/bios state buffer size
3751                                    ;                  ; pointer flags for buffer state options
3752                                    %endif
3753
3754                                    %if 1
3755                                    ; 10/12/2020
3756                                    LFB_Info:
3757 00031212 <res 10h>                            resb 16      ; Linear Frame Buffer info block
3758
3759                                    ;24/11/2020 - TRDOS 386 v2.0.3
3760                                    ; BOCHS/PLEX86 VESA VBE3 MODE INFO extension to TRDOS 386 v2 kernel
3761                                    MODE_INFO_LIST:
3762 00031222 <res 44h>                            resb 68      ; mode + 66 byte VESA vbe3 mode info (4F01h)
3763                                    %endif
3764
3765                                    ; 05/01/2021
3766 00031266 ??                        ufont:      resb 1 ; (VGA graphics) user font flags
3767                                                       ; bit 7 - permission flag for int 31h
3768                                                       ; bit 4 - 8x16 user font ready/loaded flag
3769                                                       ; bit 3 - 8x8 user font ready/loaded flag
3770                                                       ; bit 1 - select 8x16 user font (sysvideo)
3771                                                       ; bit 0 - select 8x8 user font (sysvideo)
3772 00031267 ??                                   resb 1 ; 19/01/2021
3773                                    ; 17/01/2021
3774 00031268 ??                        srvsf:             resb 1 ; 'save restore video state' permission flag
3775                                    ; 18/01/2021
3776 00031269 ??                        srvso:             resb 1 ; video state buffer save/restore option
3777 0003126A ????????                  VideoStateID:      resd 1 ; used to verify state saved by same prog
3778                                    ; 29/01/2021
3779 0003126E ????                      v_width:    resw 1 ; screen (display page) width
3780 00031270 ??                        v_ops:             resb 1 ; 'sysvideo' graphics data transfer option
3781 00031271 ??                        v_bpp:             resb 1 ; bits per pixels ('sysvideo')
3782 00031272 ????????                  v_mem:             resd 1 ; video memory ('sysvideo')
3783 00031276 ????????                  v_siz:             resd 1 ; video page size ('sysvideo')
3784 0003127A ????????                  v_str:             resd 1 ; window start adress ('sysvideo')
3785 0003127E ????????                  v_end:             resd 1 ; window end (end+1) adress ('sysvideo')
3786                                    ; 31/01/2021
3787                                    ; 01/01/2021
3788                                    ;maskbuff:   ;resd 1 ; user's bitmask buffer addr ('sysvideo')
3789 00031282 ????????                  maskcolor:  resd 1 ; VGA/SVGA pixel mask color ('sysvideo')
3790                                    ; 27/02/2021
3791 00031286 ????????                  pixcount:   resd 1 ; pixel count ('sysvideo' window ops)
3792                                    ; 02/02/2021
3793 0003128A ????????????????          buffer8:    resd 2 ; 8 bytes small buffer for 'sysvideo'
3794
3795                                    bss_end:
3796
3797                                    ; 27/12/2013
3798                                    _end:  ; end of kernel code              _end:  ; end of kernel code
```